Fall 2007

GlySpy: A software suite for assigning glycan topologies from sequential mass spectral data

Anthony Lapadula
University of New Hampshire, Durham

GLYSPY

A SOFTWARE SUITE FOR ASSIGNING GLYCAN TOPOLOGIES FROM SEQUENTIAL MASS SPECTRAL DATA

BY

ANTHONY LAPADULA
B.S., University of New Hampshire, 1990
M.S., University of New Hampshire, 1991

DISSERTATION

Submitted to the University of New Hampshire
in Partial Fulfillment of
the Requirements for the Degree of

Doctor of Philosophy
in
Computer Science
September 2007

This dissertation has been examined and approved.

Dissertation Director
Philip J. Hatcher
Professor of Computer Science

David J. Ashline
Post Doctoral Research Associate
Department of Chemistry

R. Daniel Bergeron
Professor of Computer Science

Vernon N. Reinhold
Professor of Molecular Biochemistry,
Biochemistry, and Chemistry

Ted M. Sparr
Professor of Computer Science

July 11, 2007
Date

DEDICATION

To Kathleen, Elena, and Anna – my sweethearts

ACKNOWLEDGEMENTS

Colleagues

I am deeply indebted to the many friends and colleagues who contributed to this work. I wish to thank my committee for their time and effort. Many people in Dr. Reinhold's lab provided help along the way, including Hailong Zhang, Justin Prien, Joe Gieser, Suddham Singh, and Kevin Bullock, but I would like to give special recognition to those who spent the most time teaching me the tricks of the trade: Vernon Reinhold, David Ashline and Andy Hanneman. Last, but certainly not least, I want to thank Phil Hatcher for his continued friendship and support, and his near infinite patience in reviewing early drafts of this work.

Reprint Permissions

Figure 19 from (Tang 81) is reproduced by permission of Oxford University Press. Figure 21 from (Goldberg 32) is reproduced with permission of both Wiley-VCH Verlag GmbH & Co KGaA and David Goldberg, who very kindly provided an electronic copy of the illustration. Material from (Lapadula 54) and (Ashline 6) and Figure 20 from (Tseng 82) are reproduced in part by permission of the American Chemical Society. Copyright 2005, 2007, and 1999, respectively, American Chemical Society. Figure 18 from (Ethier 25) is reproduced with permission of John Wiley & Sons Limited. Copyright 2002 John Wiley & Sons Limited.

Funding

Funding provided by the National Institutes of Health NCRR BRIN P20 RR16459 and NIGMS R01 GM54045 is gratefully acknowledged.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SPECTRA

LISTINGS

ABSTRACT

GLYSPY

A SOFTWARE SUITE FOR ASSIGNING GLYCAN TOPOLOGIES FROM

SEQUENTIAL MASS SPECTRAL DATA

by

Anthony Lapadula

University of New Hampshire, September 2007

GlySpy is a suite of algorithms used to determine the structure of glycans. Glycans, which are orderly aggregations of monosaccharides such as glucose, mannose, and fucose, are often attached to proteins and lipids, and provide a wide range of biological functions. Previous biomolecule-sequencing algorithms have operated on linear polymers such as proteins or DNA but, because glycans form complicated branching structures, new approaches are required. GlySpy uses data derived from sequential mass spectrometry (MS^n), in which a precursor molecule is fragmented to form products, each of which may then be fragmented further, gradually disassembling the glycan. GlySpy resolves the structures of the original glycans by examining these disassembly pathways.

The four main components of GlySpy are: (1) OSCAR (the Oligosaccharide Subtree Constraint Algorithm), which accepts analyst-selected MS^n disassembly pathways and produces a set of plausible glycan structures; (2) IsoDetect, which reports the MS^n disassembly pathways that are inconsistent with a set of expected structures, and which therefore may indicate the presence of alternative isomeric structures; (3) IsoSolve, which attempts to assign the branching structures of multiple isomeric glycans found in a complex mixture; and (4) Intelligent Data Acquisition (IDA), which provides automated guidance to the mass spectrometer operator, selecting glycan fragments for further MS^n disassembly.

This dissertation provides a primer for the underlying interdisciplinary topics—carbohydrates, glycans, MS^n, and so on—and also presents a survey of the relevant literature with a focus on currently-available tools. Each of GlySpy's four algorithms is described in detail, along with results from their application to biologically-derived glycan samples. A summary enumerates GlySpy's contributions, which include *de novo* glycan structural analysis, favorable performance characteristics, interpretation of higher-order MS^n data, and the automation of both data acquisition and analysis.

CHAPTER 1:

INTRODUCTION

Glycans (Rademacher 70; Van den Steen 85; Various 86; Varki 88) are oligosaccharides that are conjugated to fats (lipids) and over half of human proteins (Apweiler 4), and play important roles in a wide variety of biological processes (Dwek 22; Gabius 28; Gabius 29; Ioffe and Stanley 42; Lowe and Marth 59; Stanley and Ioffe 78; Van den Steen 85; Varki 87; Varki 88). Glycans are "so ubiquitous ... that cells appear to other cells and to the immune system as sugarcoated" (Maeder 60). Unlike linear DNA and proteins, glycans can form complicated branching structures, where one monosaccharide residue may be linked to several others. These linkages also have variables such as linkage position and anomericity, resulting in astonishing numbers of theoretically possible structures (Laine 51). Because glycans cannot be amplified as DNA can, glycan sequencing technologies must operate on minute quantities of oligosaccharides, often eliminating Nuclear Magnetic Resonance (NMR) as a feasible analysis method. Structural analysis may be augmented with enzymes that cleave glycans in well-defined ways, but these methods are restricted by the limited number of available exo- and endoglycosidases (Küster 50) and by the fact that many such enzymes are not completely specific.

These issues and others have made glycobiology, the study of carbohydrates in biological processes, the focus of increasing interest and a fertile ground for bioinformatics efforts

(Marchal 61; von der Lieth 90). No single sequencing strategy has yet emerged, but the extreme sensitivity of mass spectrometry (MS) and sequential mass spectrometry (MS^n) combined with bioinformatics tools will provide significant progress toward high-throughput glycan sequencing.

Glycans are significant in a number of biological and biomedical research areas. For instance, glycans are biomarkers for various cancers (Alper 2; Dziadek and Kunz 24; Ono and Hakomori 67; Turner 84) and the principal component of new and promising vaccines for diverse cancers (Lo-Man 58), viruses (Dwek 23), and bacteria (Ada and Isaacs 1; Gaucher 30; Muhlecker 64). They drive parasite-host (Hokke and Deedler 39; Khoo and Dell 47; Nyame 66) and microbe-host (Hooper and Gordon 40) interactions, as well as egg fertilization (Hedrick and Nishihara 38; Mozingo and Hedrick 63; Tseng 83) and protein folding (Parodi 68). They are crucial to drug development efforts (Dove 21; Koeller and Wong 48; Walsh 92) and are involved in allergic (Huby 41) and inflammatory responses (Kannagi 46). Defective glycan metabolism manifests as Congenital Disorders of Glycosylation, Gaucher, Fabry, Tay-Sachs, and Sandhoff diseases, among others (Butters 14; Dwek 23; Jaeken and Matthijs 43; Jeyakumar 44; Platt 69; Vosseller 91). Research in these and related areas is hindered by the lack of effective glycan sequencing tools and methods (Stephan 79).

The UNH Glycomics Center, formerly the Center for Structural Biology, has pioneered techniques to establish the topology (branching and linkage) of glycans using MS^n (Ashline 5; Ashline 7; Hanneman and Reinhold 33; Hanneman 34; Hanneman and Reinhold 35; Lapadula 54; Reinhold 71; Reinhold 72; Reinhold 73; Sheeley and Reinhold 75; Singh and Reinhold 76; Singh 77; Zhang and Reinhold 95; Zhang 96; Zhang 97), and other methods have been developed internationally (Cancilla 16; Harvey 36; Harvey 37; König and Leary 49; Viseux 89; Weiskopf 93; Xie 94).

For several years, the author has been developing GlySpy (Lapadula 52; Lapadula 53; Lapadula 54) at the UNH Glycomics Center to automate much of the glycan structural analysis currently performed manually. GlySpy uses data derived from sequential mass spectrometry (MS^n), in which a precursor molecule is fragmented to form products, each of which may then be fragmented further, gradually disassembling the glycan. GlySpy resolves the structures of the original glycans by examining these disassembly pathways.

GlySpy consists of four major, interrelated components:

1) **OSCAR** (the *O*ligosaccharide *S*ubtree *C*onstraint *A*lgo*r*ithm), which accepts analyst-selected MS^n disassembly pathways and produces a set of plausible glycan structures;

2) **IsoDetect**, which reports the MS^n disassembly pathways that are inconsistent with a set of expected structures, and which therefore may indicate the presence of alternative isomeric structures;

3) **IsoSolve**, which attempts to assign the branching structures of multiple isomeric glycans found in a complex mixture; and

4) **Intelligent Data Acquisition (IDA)**, which provides automated guidance to the mass spectrometer operator, selecting glycan fragments for further MS^n disassembly.

These tools range in capability from OSCAR, which requires precisely-selected input from a skilled analyst, through IsoDetect, which performs in a more autonomous fashion, to IsoSolve and IDA, which execute with virtually no human guidance. Each tool has its own strengths and limitations, as discussed throughout this document. The algorithms are presented in detail, along with results acquired from biologically-derived glycan samples, as opposed to simpler synthetic test cases preferred by some existing tools. Relevant MS^n spectra are given in

APPENDIX A: SELECTED EXPERIMENTAL SPECTRA. OSCAR uses over 50 inference rules to efficiently eliminate candidate glycan structures from consideration. A selection of these is found in APPENDIX B: SAMPLE OSCAR INFERENCE RULES. These rules, nearly unanimously, operate on the abstract tree structure inherent in glycan topology and are not heavily dependent on chemical knowledge.

To begin, this dissertation provides a primer for the underlying interdisciplinary topics (carbohydrates, glycans, sequential mass spectrometry, and so on) before moving on to a large-scale view of GlySpy's components, and then to a survey of the relevant literature with a focus on currently-available tools. Next each algorithm is presented along with results and discussion. Finally a summary of the work is given, with an emphasis on GlySpy's contributions to the fields of glycomics and computer science. These contributions include *de novo* glycan structural analysis, favorable performance characteristics, interpretation of higher-order MS^n data, and the automation of both data acquisition and analysis.

We know of no other practical approaches to inferring tree structure that depend upon examining subtree fragments derived from sequential disassembly of the original tree. Careful software engineering was required to avoid falling prey to the combinatorial aspects of this problem. Adapted techniques include aspects of blackboard systems, constraint-based systems, and guided search algorithms, and are applied to the challenging interdisciplinary topic of glycan structural analysis.

CHAPTER 2:

BACKGROUND

A complete description of carbohydrates and glycans is beyond the scope of this dissertation. What follows is largely derived from (Brooks 10; Brown 11; Campbell and Farrell 15).

2.1. Carbohydrates and Glycans

The term *carbohydrate* comes from the chemical formula satisfied by the earliest-known substances of this class: $(CH_2O)_n$, that is, one carbon for each water, or *carbo-hydrate*. Carbohydrates are the most abundant organic substances produced by living organisms, storing energy and forming structural components such as cellulose. They are formed from monosaccharide building blocks including glucose, mannose, galactose, fucose, GlcNAc, GalNAc, and Neu5Ac (Figure 1). When incorporated into a larger carbohydrate, the monosaccharides are known as *residues*.

When a carbohydrate is attached to a protein or a lipid, it is often called a *glycan*, and the combined structure is called a *glycoprotein* or a *glycolipid*. The residue attached to the protein or lipid is identified as the reducing-end residue, and will be drawn as the right-most residue in this document. The reducing-end sugar can be thought of as the root of a tree, with terminal residues constituting the tree's leaves.

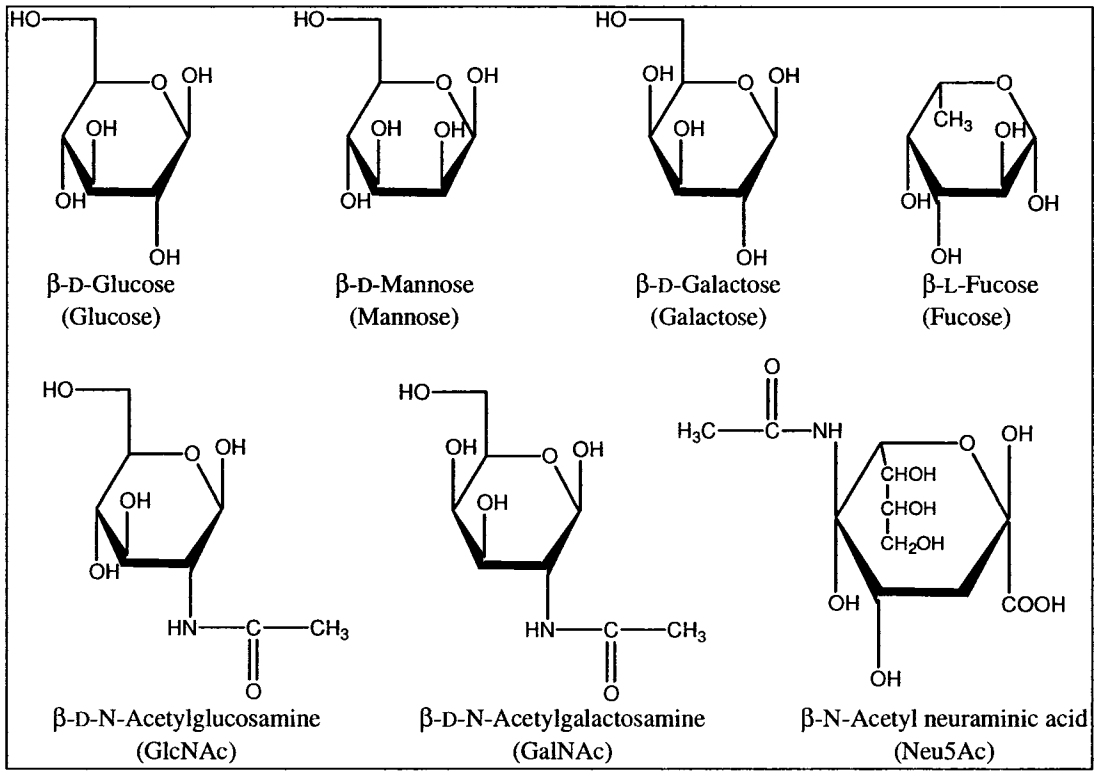


Figure 1: The monosaccharides glucose, mannose, galactose, fucose, GlcNAc, GalNAc and Neu5Ac.

The work described in this document utilizes glycans derived from a variety of sources, with special focus on these samples as purchased from Sigma-Aldrich Co., St. Louis, MO:

1) Chicken ovalbumin, the main protein in egg white (product A5503)

2) IgG, human serum immunoglobulin G (product I4506)

3) Purified type III bovine brain gangliosides, a glycosphingolipid (product G2375)

4) Fetuin, a fetal calf blood protein (product F2379)

2.2. Glycan Release, Reduction and Permethylation

The UNH Glycomics Center routinely derivatizes (chemically modifies) glycans before MS^n analysis. The exact chemical protocols used are beyond the scope of this document, but

descend from a long line of techniques, most recently (Ciucanu and Kerek 17), and are described more fully in (Ashline 5). A brief overview of these techniques may be useful.

Glycans are first *released* from their conjoiners and purified. This yields a complex mixture of oligosaccharides, and direct links back to their sources are lost. Frequently, the exposed hemiacetal bond is *reduced* with sodium borohydride to form an alditol, breaking the carbon ring of the reducing-end (root) sugar and giving it a modified mass that serves as a reference anchor during MS^n analysis. Last, the glycans are *permethylated.* Here, methylation replaces all acidic protons, in effect converting all hydroxyl groups (OH) to methoxyl groups (OCH_3, abbreviated OMe). Permethylation allows for the detection of cleavages between residues, as will be discussed in Section 2.6.3.

Figure 2 shows the results of derivatization on the monosaccharides introduced above. We establish class names to represent monomers with identical masses: **H** for hexose (glucose, mannose, and galactose); **F** for deoxyhexose (fucose); **N** for HexNAc (GlcNAc and GalNAc); and **S** for sialic acid (Neu5Ac). GlySpy supports three reduced residues derived from H, F, and N; these are designated **h**, **f**, and **n**, respectively. Reduced fucose residues (f) are quite rare and are not considered further in this report; both h and n are included in Figure 2.

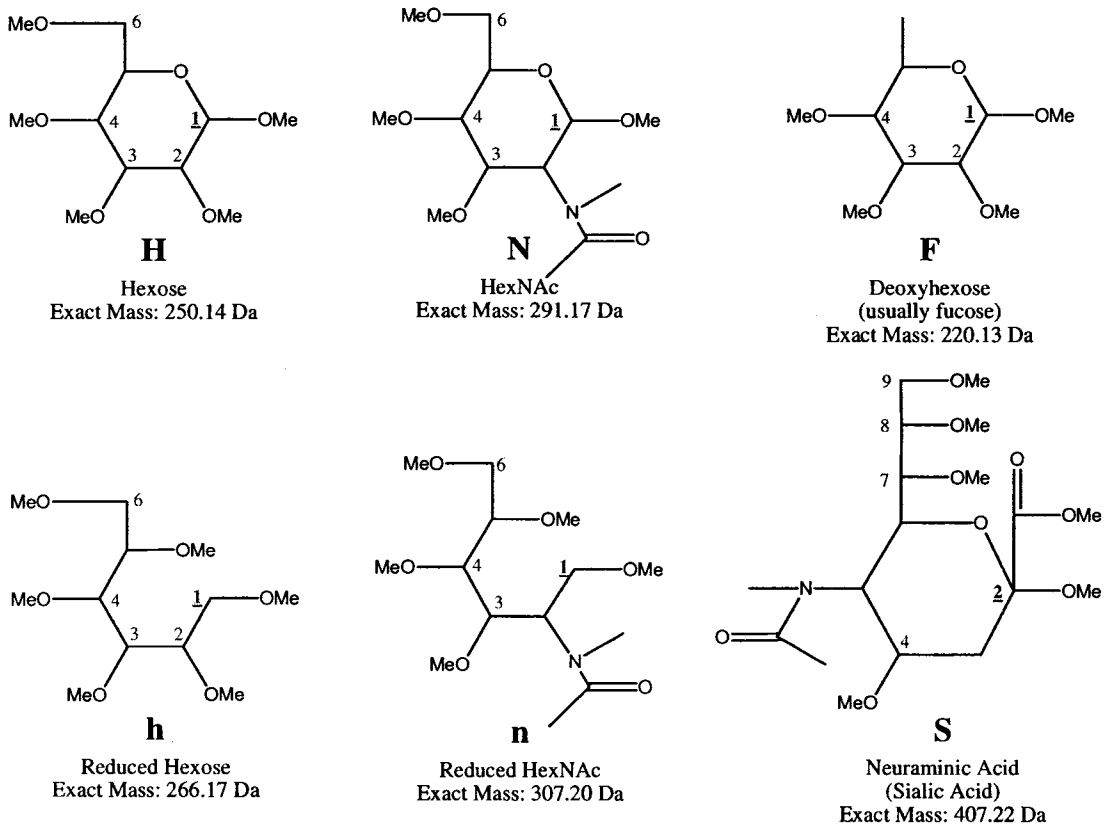


Figure 2: The permethylated monosaccharide classes H, N, F, and S, and the permethylated, reduced classes h and n. Linkage positions are numbered; anomeric carbons are highlighted.

Figure 3 shows a simplified representation of the monosaccharides from Figure 2. Notice that reduced residues are distinguished by the case of their label, not by a difference in shape. This representation is a simplification of (Nomenclature Committee of the Consortium for Functional Glycomics 65).

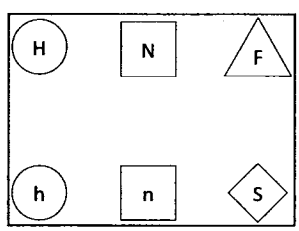


Figure 3: Simplified representation of the monosaccharide classes of Figure 2.

2.3. Interresidue Linkage and Anomericity

Monosaccharides combine to form disaccharides, trisaccharides, and so on, by forming glycosidic bonds in one of two possible stereochemical anomeric orientations, axial (alpha or α) or equatorial (beta or β). The interresidue bonds extend from the anomeric carbon (carbon 2 for sialic acid, carbon 1 otherwise) of the non-reducing-end sugar to an available position (carbons 4, 7, 8 or 9 for sialic acid; otherwise a subset of carbons 2, 3, 4, or 6) of the reducing-end sugar. The linkage positions for the supported residues are shown in Figure 2, with the anomeric carbons highlighted. Other monosaccharide residues, for example fructose, have different linkage positions, but these are outside the scope of this document.

Figure 4 shows a hypothetical trisaccharide with individual residues labeled with superscripts. Residue F^0 is terminal (a *leaf*), H^1 in internal, and n^2 is at the reducing end (the *root*). Using the linkage positions shown, we would designate this structure as F1-4H1-4n; that is, an F residue 1-4 linked to an H, which is 1-4 linked to n. As GlySpy does not attempt to compute stereochemistry, anomericity is omitted from many figures.

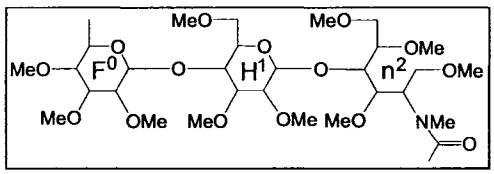


Figure 4: A hypothetical trisaccharide

2.4. *N*-Glycans and *O*-Glycans

N-linked glycans, or simply *N*-glycans, are always attached to proteins at the nitrogen atom (hence, "N") of the amide group of an asparagine amino acid residue. Importantly, they nearly always contain a trimannosyl core consisting of five residues linked in an unwavering formation: two mannoses α1-3 and α1-6 connected to a single mannose, which is β1-4 connected to an

internal GlcNAc, which is β1-4 connected to the reducing end GlcNAc. See Figure 5. Larger *N*-glycans attach additional residues to this core.

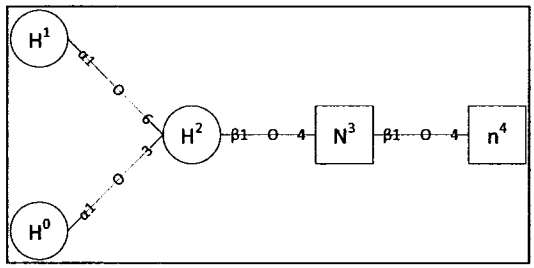


Figure 5: The five residues of the conserved *N*-linked core.

O-linked glycans, or *O*-glycans, are attached to the oxygen atom (hence, "O") of a serine or threonine amino acid. They commonly consist of from one up to perhaps a dozen residues and are often classified according to a series of common core structures, Core 1–Core 8, as shown on page 93 of (Brooks 10).

2.5. Terminology: Chemistry vs. Computer Science

The interdisciplinary nature of this work may cause confusion because of the differing terminology used in the fields of chemistry and computer science. To reduce this confusion, we define some useful synonyms. Table 1, which refers to Figure 5, defines some equivalent terms which will be used interchangeably in this document.

Table 1: Equivalent terminology from chemistry and computer science.

Chemistry	Computer Science
Glycan	*Tree*
The glycan's *residues* are H^0, H^1, H^2, N^3, n^4	The tree's *nodes* are H^0, H^1, H^2, N^3, n^4
n^4 is the *reducing-end residue*	n^4 is the *root* of the tree
H^1 is a *non-reducing-end terminal residue*	H^1 is a *leaf*
H^1 forms a *glycosidic bond* with H^2	H^1 is a *child* of H^2 (or H^2 is the *parent* of H^1)
H^2 has two *substituents*, H^0 and H^1	H^2 has two *children*, H^0 and H^1

2.6. Mass Spectrometry

2.6.1. MS and MS^n

Mass spectrometry (MS) and sequential mass spectrometry (MS^n) are well-established methods for oligosaccharide analysis. There are many types of mass spectrometers, but, very simply put, these instruments measure the mass-to-charge ratio (denoted *m/z*) of ionized (electrically charged) sample molecules. The result is a spectrum plotting *m/z* against relative abundance. See Figure 6 for a detail from such a spectrum.

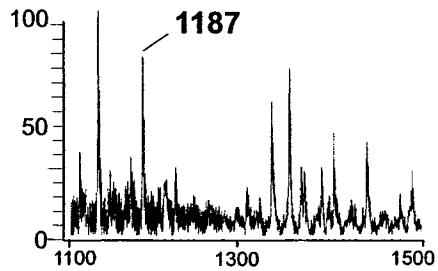


Figure 6: A portion of an MS profile spectrum showing abundance and mass, with the ion *m/z* 1187 indicated.

The full range of ionization and detection technologies available is beyond the scope of this document. Sequential mass spectrometry (MS^n), using an ion trap (IT-MS), allows the operator to select peaks ("precursor ions") from a spectrum, fragment them, and record the resulting "product ions" in another spectrum. Fragmenting a peak from the initial MS spectrum yields an MS^2 spectrum; fragmenting a peak from that yields an MS^3 spectrum; and so on.

Figure 7 shows a typical multi-step disassembly of ion *m/z* 1187.6. As we will see, this piecewise disassembly provides valuable clues at each step; OSCAR uses these clues to deduce the topology of the unfragmented glycan. In the figure, each oval represents the *spectrum* generated from fragmenting the labeled precursor ion. The MS^3 spectrum shown here was generated by selecting 1187.6 from the MS profile spectrum, fragmenting it, selecting 894.4

from that spectrum, and fragmenting it. From this spectrum, both 649.2 and 676.3 can be selected for MS^4, and so on.

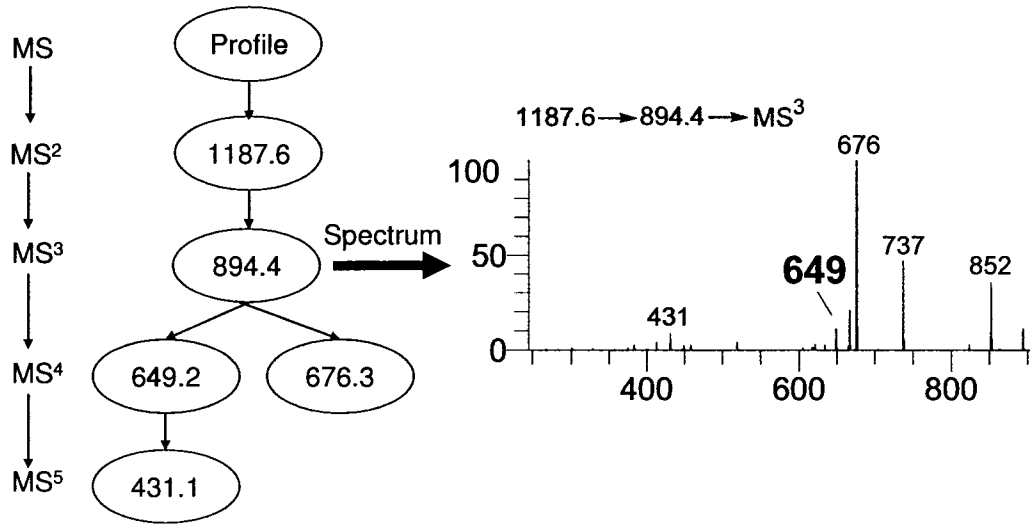


Figure 7: Two overlapping MS^n fragmentation pathways: *m/z* 1187.6 → 894.4 → 649.2 → 431.1 and *m/z* 1187.6 → 894.4 → 676.3. At right is the spectrum generated during disassembly of ion *m/z* 894.4.

2.6.2. Da vs. *m/z*

The hypothetical trisaccharide of Figure 4 has a monoisotopic mass of 685.39 Da. That is, this is the total mass in daltons if the compound were composed entirely of the most abundant isotopes of its constituent atoms. A dalton is defined to be 1/12 the mass of a carbon-12 atom.

However, this trisaccharide is uncharged and thus undetectable by a mass spectrometer. To solve this problem, we establish conditions that cause adduction of a metal cation, usually sodium (Na^+), to the compound. The mass of the sodium ion is 22.99 Da and its charge is +1. Therefore the observed *m/z* of the trisaccharide is computed as:

$$m/z = (685.39 + 22.99) / (+1) = 708.38$$

If two sodium ions had been adducted, the observed *m/z* would have been:

$$(685.39 + 22.99 + 22.99) / (+2) = 365.68$$

In this document, measurements given in Da are masses with no adducted ion, whereas *m/z* always refers to an observed ion, sodiated unless otherwise specified.

2.6.3. Inferring Topology from MS^n Data

To assign the topology of a glycan, an analyst will examine fragments generated by MS^n disassembly. Figure 8 shows a simple hypothetical example. When the ion *m/z* 708.4 (Figure 8A) is fragmented, assume that high-intensity fragment ions *m/z* 211.1, 227.1, and 316.2 are observed (Figure 8B). The most likely interpretations of these values are as follows:

- *m/z* 211.1: A terminal fucose with a 1,2-ene (a double bond between carbons 1 and 2) cleavage at the reducing-end carbon. We abbreviate this composition (residues plus cleavages) as F-(ene)'. (The trailing prime indicates that one of the scars must occur on the reducing end. Composition notation will be covered more fully in Section 3.4.)
- *m/z* 227.1: An internal hexose with both a 1,2-ene cleavage and an open hydroxyl (OH) cleavage. Composition: H-(ene)(oh)'.
- *m/z* 316.2: A reduced HexNAc residue with a single open hydroxyl cleavage: n-(oh).

This interpretation is the most plausible because the glycosidic bonds joining monomers are the most labile and where fragmentation occurs. Thus, most abundant ions will be the result of glycosidic cleavages. Cross-ring cleavages, multiple simultaneous cleavages, and other interpretations are possible as well, but these typically yield only low-intensity peaks.

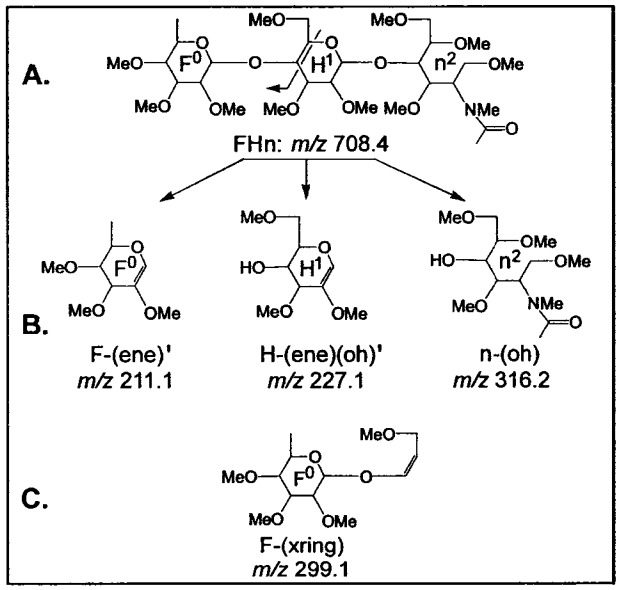


Figure 8: Fragments potentially generated by disassembly of the hypothetical trisaccharide FHn.

Because the glycans are permethylated, the fragments generated during MS^n preserve hints of their original connectivity. Specifically, the number of (ene) and (oh) scars in each composition indicate the *number* of cleavages applied to the fragment, although the original *linkage* and *identity* of the cleaved residues are not directly recorded. In this case, the observed composition n-(oh) reveals only that the n residue had a single residue connected directly to it, but not the identity of the residue. Similarly, the H-(ene)(oh)' fragment tells us that the H residue had previously been directly connected to two residues, and F-(ene)' indicates that the F residue had only a single attached residue. Combining these observations, plus the fact that a reduced n can occur only at the reducing end of a glycan, the analyst easily infers the linear FHn topology of Figure 8A.

Figure 8C shows one cross-ring fragment that might be observed: part of the H's ring is still attached to the terminal F. The mass of this cross-ring fragment reveals that F^0 is linked to either position 4 or 6 of H^1. (The linkage could just have easily been 1-6 instead of the shown 1-4; the mass of the fragment would have been identical.) Multiple cross-ring cleavages are

sometimes required to confirm a linkage assignment; often, a precise determination cannot be made.

2.6.4. Cross-Ring Fragments

Cross-ring fragments are identified by the bonds cleaved to generate the fragment and whether or not the fragment contains the anomeric carbon of the cleaved residue (Domon and Costello 20). Figure 9 shows the bond numbering for a hexose residue; all residues supported by GlySpy share this scheme.

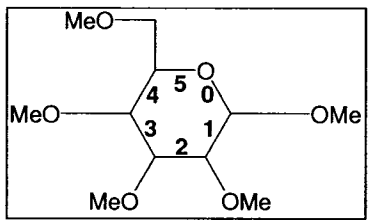


Figure 9: Bond numbering used to identify cross-ring cleavages of a hexose. Bond numbers derive from the carbon which they follow.

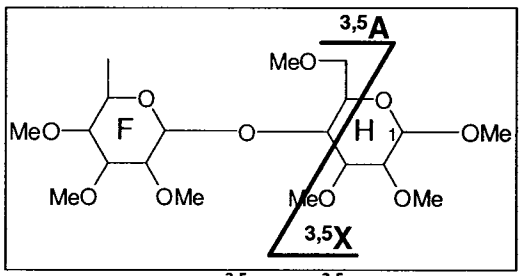


Figure 10: Complementary ^{3,5}A and ^{3,5}X cross-ring fragments.

Figure 10 shows the two fragments that would result from cleaving bonds three and five of the reducing-end hexose. The fragment without the anomeric carbon (labeled "1") is denoted the ^{3,5}A fragment; the complementary fragment is denoted ^{3,5}X. We can now see that the cross-ring fragment of Figure 8C could more precisely be described as having composition F-^{3,5}A[HNn], where the [HNn] denotes the residue classes that might have generated the cross-ring fragment. H, N, and n all share the same atomic structure at the relevant parts of the residues, and hence

any of these might have generated the fragment. Notice that F-^{3,5}A[F] is not a valid composition, as a reducing-end F residue could not produce the fragment exactly as shown—F has no OMe at carbon six.

2.7. Structural Isomers

The same monosaccharide building blocks can be linked to form a variety of structures called *structural isomers*[1]. For example, Figure 11 shows two isomeric glycans with the same composition, but GM1a is branched while GM1b is linear. One goal of this research is to successfully analyze mixtures of structural isomers.

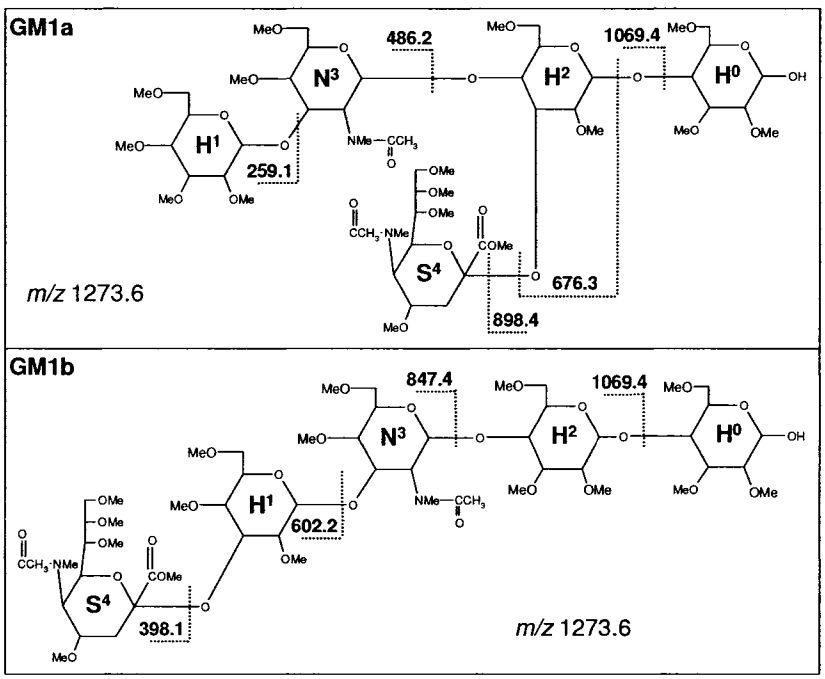


Figure 11: The isomeric glycoconjugates from the bovine brain gangliosides GM1a and GM1b. These glycans contain the same residues, but have different structures.

[1] The terms *isomer* and *isobar* are often used interchangeably, creating much confusion in the field. In this work, we adopt the terminology of (62). There, *structural isomer* (shortened to *isomer*) indicates identical atomic compositions arranged in a different structure. We extend this definition slightly to mean the *same monosaccharide residues* arranged in a different configuration, as in Figure 11. The term *isobar* will indicate *different composition of atoms* occurring at a nominal (unit) mass resolution; that is, structures or fragments with nearly identical masses but different chemical compositions.

CHAPTER 3:

GLYSPY

3.1. Overview and Goals

GlySpy is an integrated toolkit for manual, semi-automated, and automated glycan analysis. It consists of four major algorithms:

- OSCAR
- IsoDetect
- IsoSolve
- Intelligent Data Acquisition (IDA)

OSCAR and IsoDetect are used today in the Glycomics Center and have contributed to published and forthcoming reports. As such, these algorithms are relatively mature. IsoSolve and IDA are more experimental in nature and should be viewed as stepping stones toward fully automated glycan analysis. They perform quite well in situations for which they were designed but do not yet obviate the need for a human analyst. This dissertation offers assessments of all four algorithms, including cases where they perform poorly. Future work items will be identified to improve performance and accuracy.

Even today, however, these tools improve analysts' capabilities and efficiency, and may assist in the discovery of biological insights in a variety of areas. In the long term, beyond the

scope of this dissertation, it is hoped that these tools will form the basis of a high-throughput glycomics analysis platform, which will automate both data collection and structural analysis. As such, the tools should be understood as moving along the continuum from fully-manual toward fully-automated glycomics analysis.

3.2. Implementation and Performance

GlySpy was implemented as a single Microsoft Windows command-line interface tool, GlySpyCLI.exe, and consists of over 50,000 lines of C++.

Execution times, when given, were measured as the best of five consecutive executions of the given test. Each test was executed at an elevated priority level (process level ABOVE_NORMAL_PRIORITY_CLASS, thread level THREAD_PRIORITY_ABOVE_NORMAL). This protocol minimized the timing variations created by a multi-tasking operating system, disk caches, and so on. The tests were performed on a Dell E1705 laptop with an Intel T7200 Centrino Core2 Duo CPU running at 2.0 GHz. The system had 2 GB of system memory and ran Microsoft Windows Vista Home Premium.

3.3. Algorithm Integration and Interaction

Figure 12 shows an extremely high-level view of GlySpy, with emphasis on the input and output of its four algorithms. We see that, typically, a prepared glycan sample is infused by an Advion TriVersa NanoMate into an LTQ mass spectrometer. In manual operation, the analyst collects data and chooses the input to provide to OSCAR, IsoDetect or IsoSolve. In automated mode, the Intelligent Data Acquisition module collects spectra, which can then be passed to IsoSolve for topology analysis.

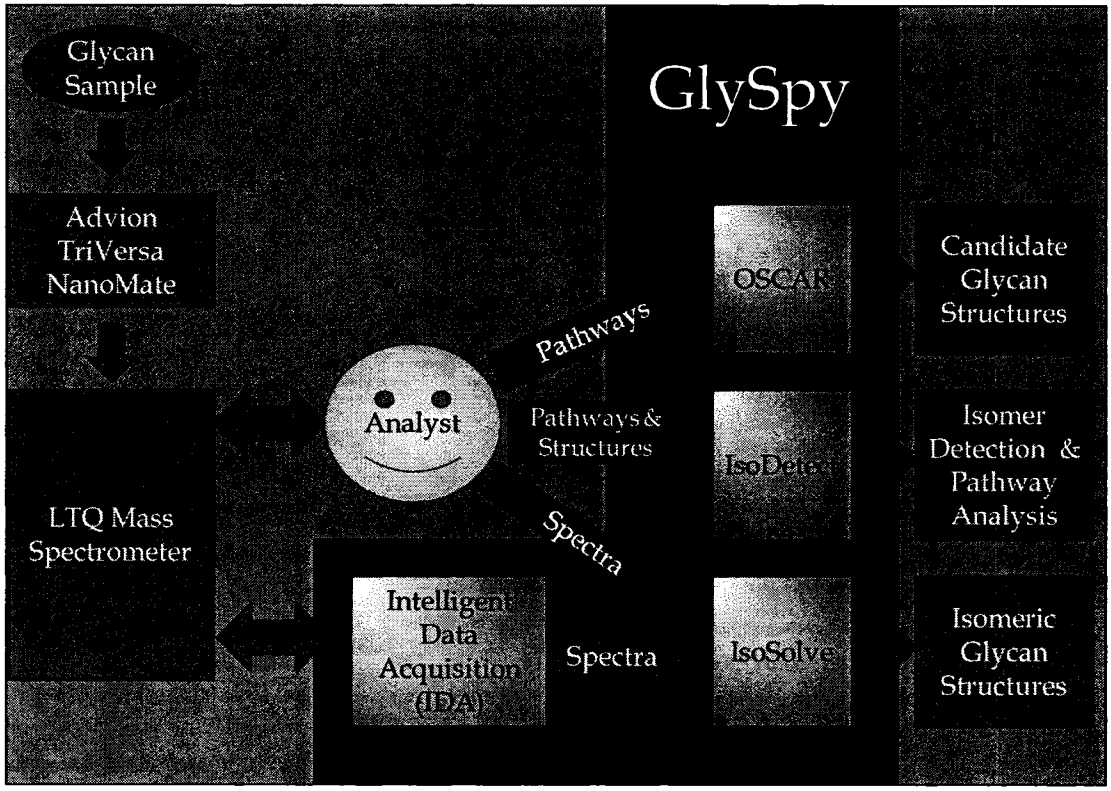


Figure 12: A high-level system diagram showing the input and output of GlySpy's four main algorithms.

Because the four GlySpy algorithms are all part of the same executable, they can communicate extremely efficiently. For example, although not shown in Figure 12, IsoDetect, IsoSolve and IDA all invoke OSCAR to perform their higher-level tasks. These interactions are covered as we describe each algorithm.

3.4. Composition Notation

Residue compositions are given as residue counts paired with scars. For example, H_4N_2n represents a composition of four hexoses, two HexNAcs, and one reduced HexNAc.

As mentioned briefly in Section 2.6.3 and Figure 8, scars are denoted by (oh) and (ene) modifiers, each of which may be modified by a count. A trailing prime (') indicates that exactly one of the scars must occur on the reducing end; if no trailing prime is present, all of the scars are located on the non-reducing end. A few examples:

- H-(oh) represents a single hexose with one (oh) scar on the non-reducing end.
- HN-$(oh)_2$ represents a Hex-HexNAc dimer, which jointly contains two (oh) scars, neither of which is on the reducing end.
- H_3-(ene)(oh)' represents a hexose trimer with both an (ene) and an (oh) scar, one of which must be a reducing-end scar, and the other a non-reducing-end scar.

Note that compositions, with the exception of the prime indicator, do not directly imply structure. For example, HN-$(oh)_2$ makes no claim as to the location of the two non-reducing-end scars. They may both be on the H, or both on the N, or split with each residue having a single scar. However, the lack of a prime indicates that the reducing end of this dimer is unscarred.

Also, note that subscripts denote the number of monomers in an ion composition (e.g., H_2 means two hexoses) and superscripts identify particular residues (H^2 means the hexose with index 2).

3.5. Composition Database

GlySpy maps masses to possible compositions via a glycan fragment database, which is built by a stand-alone program also written by the author. This program, called MakeDB, creates the database file in under 13 seconds. The database file is 52,546,296 bytes in size, and contains 6,701 entries for unfragmented glycans and 2,182,614 entries for glycan fragments. To maintain this reasonable size, some limitations are placed upon the glycans and fragments contained in the database:

- Each monomer type is limited by a maximum count, as determined by a review of reported structures: H = 10, F = 4, N = 10, S = 4, h = 1, f = 1, n = 1.
- Each composition may have at most one reduced residue (h, f, or n).

- Unfragmented glycans are limited to 15 residues and a sodiated mass of 4000 Da.
- A maximum of five scars may be present on any fragment, with all combinations of up to five (oh) scars and three (ene) scars represented.
- A total of 73 different cross-ring cleavages are supported[2].

The glycans and glycan fragments accepted by GlySpy are obviously limited to those that meet the above restrictions. In practice, though, these restrictions are generous and do not seem to be limiting. In extreme cases, an expanded version of the database could easily be generated.

3.6. Shared Options and Parameters

Because GlySpy is a single executable, its various commands share many user-selectable options. Commands that begin with a dash, e.g., -ErrTol, are global options that can be given on the GlySpy command line or on any line of input. Other options such as NoCrossRing are specific to a set of commands.

3.6.1. Shared Global Options

3.6.1.1 The –ErrTol Global Option

The –ErrTol switch gives an error tolerance in Da. When an experimental mass is used to retrieve possible compositions, all compositions in the mass range [mass - ErrTol, mass + ErrTol]

---

[2] Many of these 73 cross-ring cleavages are closely related to one another. For example, the ^{3,5}A fragment shown in Figure 8C on page 15 is still methylated at carbon 6 (labeled MeO). GlySpy's database also supports two alternate versions of this cross-ring fragment, one with an (oh) scar and one with an (ene) scar at this position. All cross-ring fragments that contain multiple linkage positions generate multiple scarred cross-ring fragment variants such as these.

are considered. The default is 0.5 Da, and so an experimental mass of 500 Da would match all compositions with masses in the range [499.5, 500.5].

3.6.1.2 The –NLinked Global Option

When the –NLinked global option is given, GlySpy will only consider structures that embed the *N*-linked core motif H_3Nn (Figure 5). The structures will have all interresidue linkages assigned as well. This option may be given when the analyst is investigating the linkage of an *N*-glycan and wishes to assign residues to the 3- or 6-branch of the *N*-linked core.

3.6.1.3 The –NLinkedBranching Global Option

The –NLinkedBranching option is similar to –NLinked with the exception that the interresidue linkages are not specified. This option is used when the analyst is investigating branching topology only, and is not concerned with linkage assignments.

3.6.1.4 The –ReducingEndResidue Global Option

The –ReducingEndResidue option specifies which residues are eligible to be the reducing-end sugar of suggested structures. The supported option values are shown in Table 2. The default is `-ReducingEndResidue any`. Many examples in this work use `-ReducingEndResidue reduced`.

Table 2: Legal values for the –ReducingEndResidue option.

Value	Selected Residue Types
`any`	Any of HFSNhfn
`unreduced`	Any of HFSN
`ueduced`	Any of hfn
Subset of `HFSNhfn`	Selected residues, for example: `-ReducingEndResidue hn`

3.6.2. The NoCrossRing Shared Command Option

Many specific commands share a common NoCrossRing option. This option forces the command to interpret a given disassembly pathway as containing glycosidic cleavages only. This is useful when the analyst explicitly wishes to exclude the possibility of cross-ring cleavages, even though fragments in the pathway may have cross-ring masses that fall within the error tolerance window. The NoCrossRing command leads to much faster execution times and increased structure specificity, and is used extensively in this work.

3.6.3. The Pathway Shared Command Parameter

Many commands accept an *m/z* disassembly pathway as an argument. For example, **`LabelPathway 1636.8_914.4_710.3_506.2_316.2`** displays possible compositions for each ion in the pathway *m/z* 1636.8 → 914.4 → 710.3 → 506.2 → 316.2.

Each ion in the pathway may be annotated with a charge state, given as **x*n***. If no charge state is given +1 is assumed. For example, **`LabelPathway 1141.6x2_1012.0x2_1537.0`** displays compositions for each ion in the pathway, with the first two *m/z* values assigned a charge state of +2 and the last ion assigned +1.

3.7. Structure Notation (Linear Code)

It is often convenient to represent a glycan structure using text instead of a diagram. GlySpy's representation is based upon (Nomenclature Committee of the Consortium for Functional Glycomics 65). In this linear code, reading from left-to-right moves from the non-reducing-end of the glycan to the reducing end, and so the final monomer listed is the reducing-end residue. Parentheses are used to designate branching.

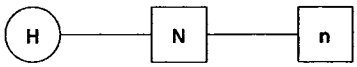
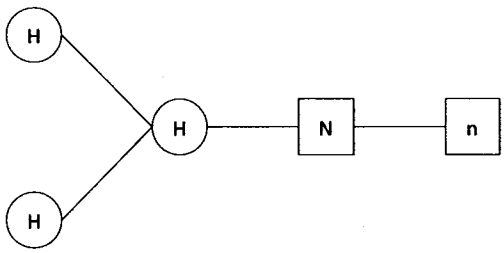
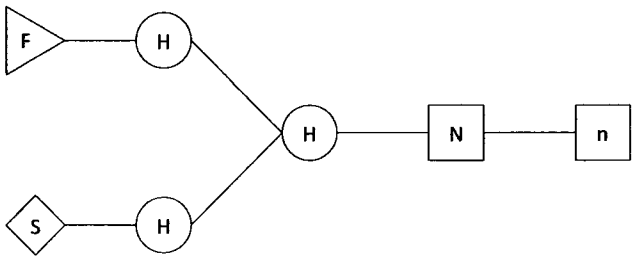
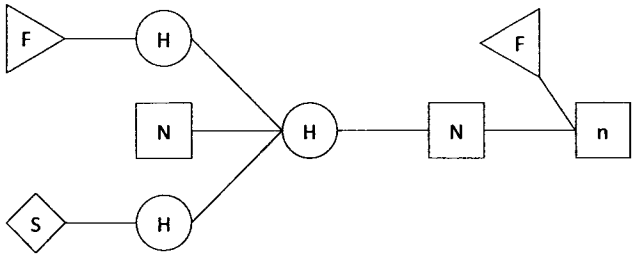
#	Hypothetical Topology	Linear Code
1		HNn
2		H(H)HNn
3		FH(SH)HNn
4		FH(SH)(N)HN(F)n

Table 3: Increasingly complex glycan topologies and their corresponding linear codes.

Table 3 shows a series of hypothetical glycan topologies along with the linear code for each. As residues are added, the topology's complexity increases. In this example, n is always the reducing end residue (or, correspondingly, the root of the tree). Topology 1 shows that linear glycans require no parentheses in their linear code, because, of course, they are not branched. Topology 2 show how a simple branch is represented in the linear code: One of the branches is parenthesized, but the other is not. (In our notation, the choice of which branch to parenthesize is arbitrary; other similar notations specify complex rules to generate canonical representations.) Topology 3 shows that branches can themselves contain linear components,

and so FH and (SH) represent the two non-reducing-end linear sequences. Topology 4 shows how additional branching is represented. Here the right-most H residue has three branches, represented as FH, (SH), and (N) in the linear code. Similarly, we see a reducing-end fucose-substituted n, represented (F)n.

The simple five residue *N*-linked core (topology 2 in Table 3, and which is described in greater detail in Figure 5 on page 10) is represented **H(H)HNn**. Optional interresidue linkages may be given as well, yielding **H6(H3)H4N4n**. A more verbose form is available, where the anomeric carbon that originates the glycosidic bond is also listed: **H1-6(H1-3)H1-4N1-4n**. Finally, alpha/beta anomericity may also be included: **Ha1-6(Ha1-3)Hb1-4Nb14n**. For *N*-linked structures, the user must indicate each core residue by applying a prime: **H'(H')H'N'n'**. If the reducing end of the glycan contains a scar, **-(oh)** or **-(ene)** may be appended.

As further examples, consider GM1a and GM1b from Figure 11 on page 16. GM1a is written as **HN(S)HH-(oh)** for branching alone, and as **H3N4(S3)H4H-(oh)** when linkage is given. Similarly, the linear GM1b is written as either **SHNHH-(oh)** or **S3H3N4H4H-(oh)**.

Note that linkage designators are neither subscripted nor superscripted, avoiding possible confusion with monomer quantities or indices, respectively.

The linear code used in this document will omit optional components not relevant to the particular algorithm being discussed. For example, none of GlySpy's algorithms consider anomericity, and so **a/b** will always be eliminated.

3.8. Known Limitations

All GlySpy algorithms are limited by the compositions found in the underlying fragment database. Glycans with other compositions will not be accepted.

OSCAR supports multiply-charged ions, but the analyst is responsible for assigning correct charge states. The remaining algorithms (IsoDetect, IsoSolve, and IDA) accept only singly-charged ions. Due to limitations of the Thermo LTQ ion trap, this will restrict these three algorithms to processing glycans with observed masses under 2000 Da. Future work is envisioned to more fully support multiply-charged ions.

OSCAR supports analysis of cross-ring cleavages, where the appropriate ions have been selected by the analyst, and computes interresidue linkages. The remaining algorithms focus solely on glycosidic cleavages and therefore compute only branching topologies.

All algorithms are capable of processing both *N*- and *O*-linked glycans. However, the glycans must be fully permethylated; acetylated or phosphorylated glycans are not supported.

3.9. Reported Glycan Structures

Glycans examined in this document come from the well-characterized sources of bovine brain gangliosides, fetuin, IgG, and ovalbumin. In many cases, we compare the structures revealed by GlySpy with those reported in the literature. For convenience, we list the reported structures in Table 4. Reported IgG structures are taken from Table II of (Butler 13), ovalbumin from (Harvey 37), and GM1a/GM1b from (Svennerholm 80).

Source	*m/z*	Composition	Branching Topology (Multiple if Isomers)
Bovine Brain Gangliosides	1273.65	H_3NS-(oh)'	GM1a: HN(S)HH-(oh) GM1b: SHNHH-(oh)
Fetuin	3618.8	$H_6N_4S_3n$	SHN(SHN)H'(SHNH')H'N'n'
IgG	1606.83	H_3FN_2n	NH'(H')H'N'(F)n'
	1636.84	H_4N_2n	HNH'(H')H'N'n'
	1677.87	H_3N_3n	NH'(NH')H'N'n'
	1810.93	H_4FN_2n	HNH'(H')H'N'(F)n
	1851.96	H_3FN_3n	NH'(NH')H'N'(F)n'
Ovalbumin	1187.61	H_3Nn	H(H)HNn
	1636.84	H_4N_2n	NH'(HH')H'N'n'
	1677.87	H_3N_3n	NH'(N)(H')H'N'n'
	1922.99	H_3N_4n	N(N)H'(N)(H')H'N'n' NH'(NH')(N)H'N'n'

Table 4: Structures reported in the literature for the glycans examined in this work. These are collected here to allow for comparison to GlySpy's results.

CHAPTER 4:

COMPARISONS TO RELATED WORK

To my knowledge, no computational tools have been developed to support an MS^n glycan sequencing strategy. Many tools use MS^2 (also known as MS/MS or tandem MS), where a single fragmentation spectrum is collected, and others use simply the MS mass to infer composition or even structure. Here existing tools are reviewed to provide some sense of the current state of the art. Because GlySpy's capabilities are built on OSCAR, we begin with high-level comparisons between this algorithm and currently available tools.

OSCAR is distinct from other currently available glycan sequencing software. For example, unlike the computer programs StrOligo (Ethier 25; Ethier 26), GlycoMod (Cooper 18) and Cartoonist (Goldberg 32), OSCAR does not encode any biologically-based structural restrictions. In other words, OSCAR does not apply constraints inferred from reported glycans or presumed biosynthetic pathways.[3] As such, OSCAR is able to assign novel glycan topologies. This architecture also enables straightforward future coverage of additional monosaccharides.

---

[3] "Biosynthetic pathways" refers to the sequence of steps that take initial reactants to final products, including the enzymes active at each step. We write that these pathways are merely "presumed" as much here remains to be discovered. For example, the pathways by which cancerous cells synthesize distinctive glycans are little understood.

Another important distinction is that GlycoMod and Cartoonist use only a composition mass, and StrOligo, GLYCH (Tang 81) and GlycosidIQ (Joshi 45) are limited to MS/MS spectral data. In contrast, OSCAR uses higher-order MS^n and interprets each product ion in the context of its precursor, and so the location of each fragment within the full glycan is assigned with greater confidence. This hierarchical precursor-product relationship is a fundamental advantage of higher-order MS^n analysis.

Importantly, OSCAR is a *de novo* algorithm, meaning it proceeds from first principles. It does not attempt to match an unknown glycan sample against a database of known glycans, as KCaM (Aoki 3) does against the KEGG and CarbBank databases or as GlycosidIQ does against the GlycoSuiteDB database (Cooper 19). OSCAR also does not attempt to match known structural motifs, as do GlycoMod and Tseng et al.'s catalog-library method (Tseng 82; Xie 94). OSCAR examines only MS^n ions in order to propose structures, and so it is free to propose novel topologies.

Next we examine a representative sampling of currently available tools in greater detail.

4.1. GlycoSuiteDB

Several attempts have been made to construct database repositories of glycan structures. Perhaps the best known of these is GlycoSuiteDB (Cooper 19), a commercial system available from Proteome Systems Limited (Sydney, Australia) at http://www.glycosuite.com. GlycoSuiteDB is an annotated and curated database that, as of Release 8.0, contains 9436 entries from 864 references and 245 species. Of the 3238 unique glycan structures, 1851 are completely characterized.

GlycoSuiteDB can be searched in a variety of ways, including composition, structure, biological source, and more. See Figure 13. Selecting "disease" and "composition" brings you to

the form shown in Figure 14. Here the user selects the disease(s) of interest along with ranges of monosaccharide residues (e.g., 0-5 Hex and 1-3 HexNAc), and is presented with a list of matching records.

The value of databases is that they are searched easily and flexibly. GlycoSuiteDB adds the important aspect of being professionally curated. However, no database can address the issue of new, unreported structures, and this is why GlySpy focuses on the area of *de novo* analysis.

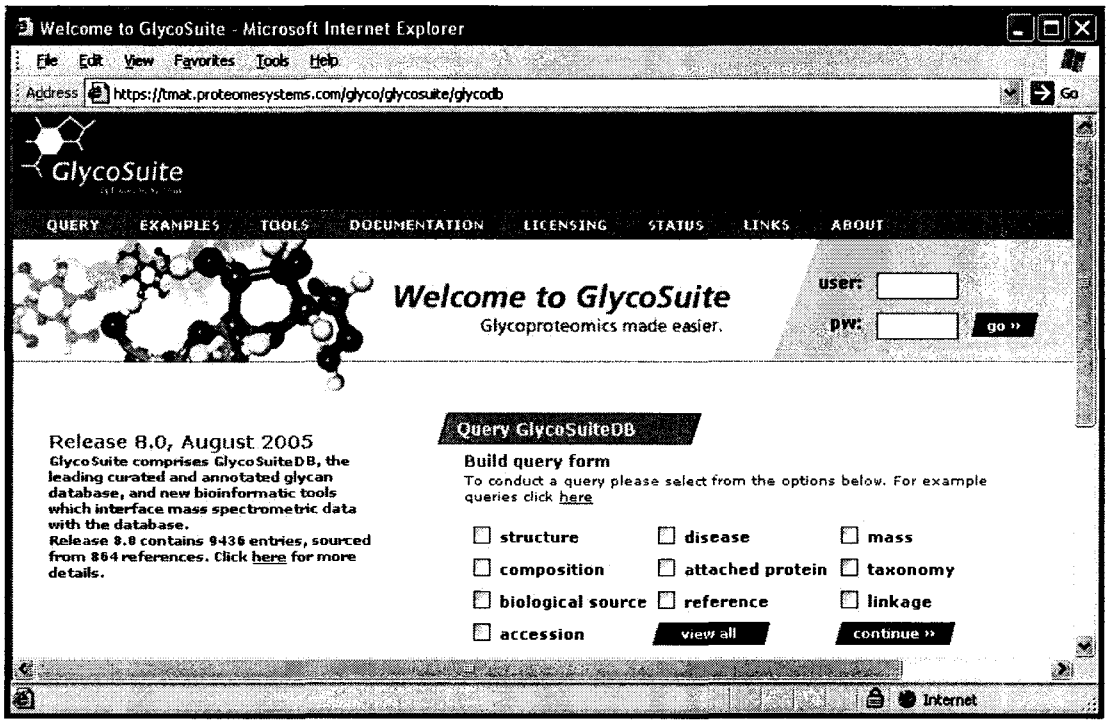


Figure 13: The GlycoSuiteDB query form.

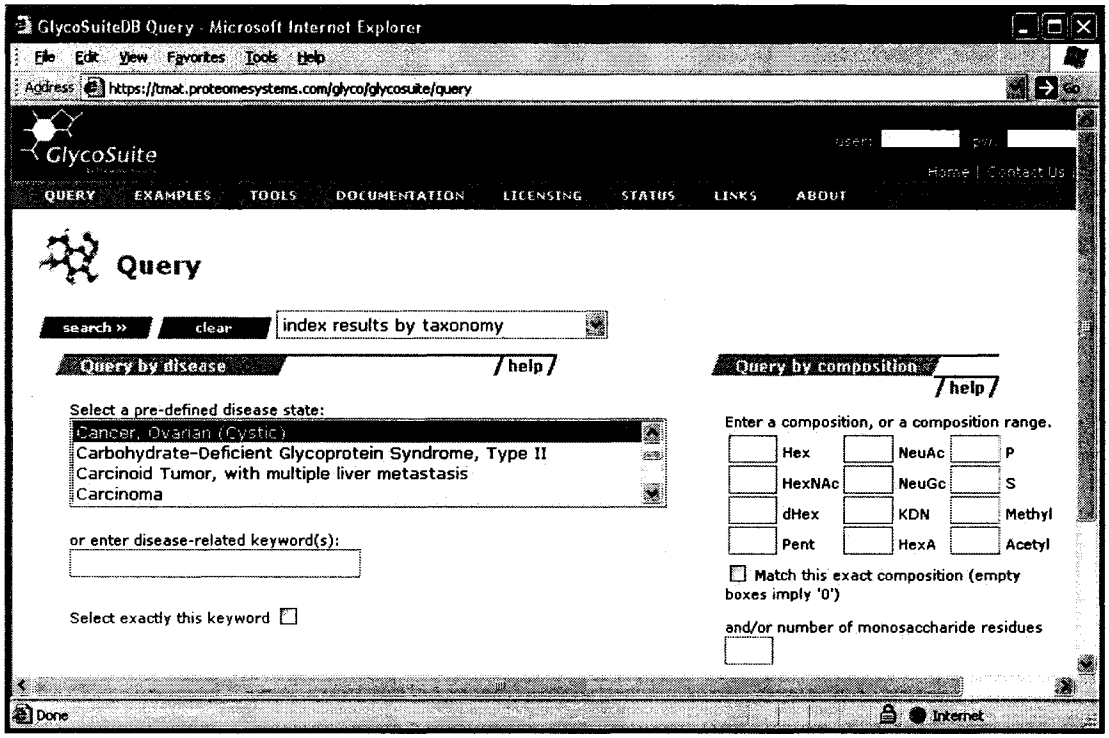


Figure 14: GlycoSuiteDB query by disease and composition.

4.2. GlycosidIQ

GlycosidIQ (Joshi 45), also available at http://www.glycosuite.com, is a commercial MS^2 glycan fingerprinting tool built upon the GlycoSuiteDB database. Its operation is split into three parts: fragmentation, matching and scoring.

First, each glycan structure in GlycoSuiteDB is presented to an *in silico* fragmentation algorithm. This algorithm creates all fragments possible given one or two glycosidic cleavages or one glycosidic cleavage plus one cross-ring cleavage. The resulting fragment mass list is saved and associated with the parent structure.

Next GlycosidIQ uses each mass on an experimental MS^2 spectrum as a search key into the database of fragment mass lists. Each structure has an associated count that records the

number of times it was matched by an experimental mass. The set of structures with a non-zero hit count are then passed to the next phase, scoring.

Here, each candidate structure is evaluated against the experimental spectrum, with penalties imposed for each feature of the structure that is unsupported by the experimental data. A second scoring scheme is used, where the high intensity peaks are given greater weight than low intensity peaks. A combined score is used to rank the candidates for presentation to the user.

The authors note that it was not possible to generate all theoretical fragments due to "computational storage and querying limitations." As such, they manually identified "specific fragments that were common to many spectra ... and manually added to the theoretical fragment database."

The results returned by GlycosidIQ are of course limited by the contents of GlycoSuiteDB. The authors claim that, when challenged with structures missing from the database, the tool returns structures that are very similar to the desired glycan. It is, however, unclear how the user would determine if the highest-ranked candidate is truly correct, or merely a close analog.

4.3. GlycanMass

GlycanMass, a tool hosted at http://www.expasy.org/tools/glycomod/glycanmass.html by the Swiss Institute of Bioinformatics, calculates the mass of a glycan given its monosaccharide composition and derivatization (underivatized, permethylated, or peracetylated). For example, selecting a permethylated glycan with composition $Hex_3HexNAc_2$ (see Figure 15) returns a monoisotopic mass of 1148.5939. GlycanMass does not offer support for reduced glycans or for charge adducts such as Na^+.

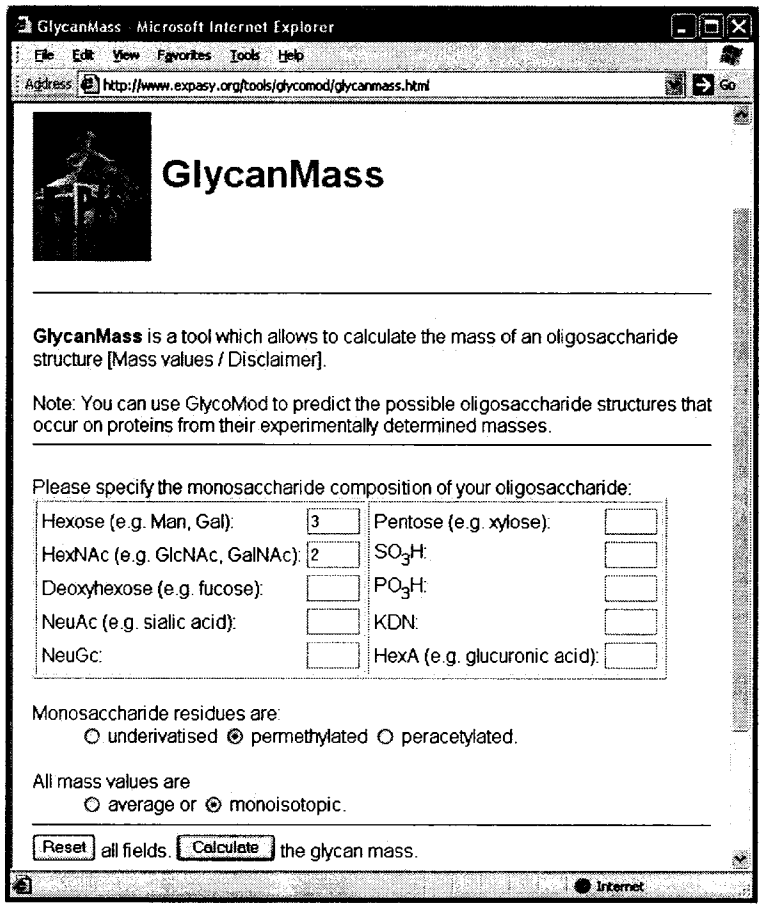


Figure 15: The GlycanMass web tool.

4.4. GlycoMod

GlycoMod (Cooper 18), available at http://www.expasy.org/tools/glycomod/, accepts an experimental mass, adduct, type of glycan (*N*-linked or *O*-linked), derivatization (underivatized, permethylated, or peracetylated), and a range for each monosaccharide type. It then produces a report listing the possible compositions for the given mass, along with a link to the matching GlycoSuiteDB database record. Figure 16 shows a portion of the input to search for a sodiated, permethylated, reduced *N*-glycan with mass 1187.7 Da. Figure 17 shows the unsurprising output that identifies this glycan as having the composition of the *N*-linked core.

GlycoMod accepts a list of experimental masses, but, importantly, these masses are all unfragmented MS^1 masses. That is, each listed ion represents a different intact glycan; the tool

merely repeats the mass-to-composition analysis for each. In contrast, OSCAR accepts multiple MS^n fragmentation pathways to assign topology, using the relationships between precursor and product ions to infer structural constraints.

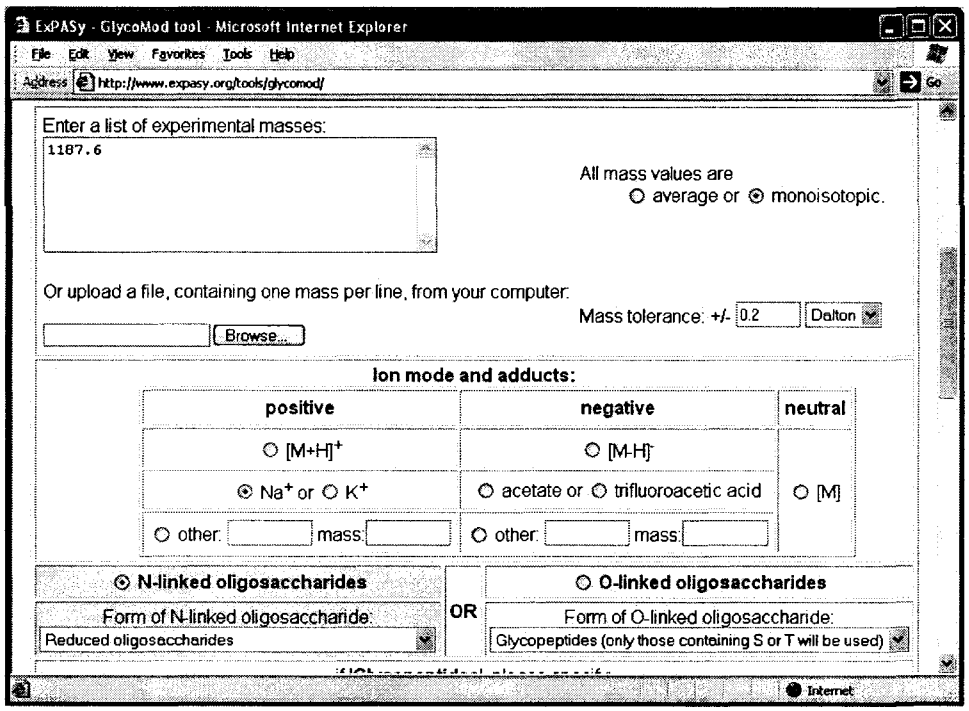


Figure 16: A section of the input page for GlycoMod.

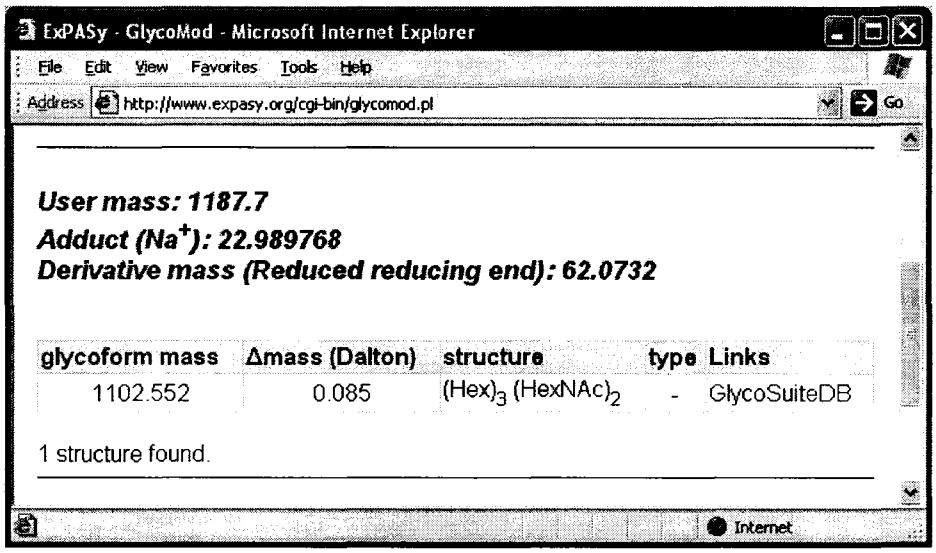


Figure 17: Sample GlycoMod output.

4.5. StrOligo

StrOligo (Ethier 25; Ethier 26) is a set of subroutines applied in series to MS^2 spectra to analyze PMP-derivatized *N*-glycans. First, isotopic peak envelopes are identified and reduced to a single monoisotopic peak. StrOligo accomplishes this by fitting the experimental envelopes to simulated envelopes generated from presumed compositions at the given mass. When successful, the isotopic peaks are merged with the monoisotopic peak, enhancing the monoisotopic peak and simplifying the spectrum.

From the remaining peaks, StrOligo examines all pairs of peaks to identify possible single and double residue losses, and this information is encoded into a relationship tree. See Figure 18, taken from (Ethier 25). Although the authors recognize that this tree may contain inaccurate relationships, they maintain that the true relationships in the tree will dominate. It is unclear what the tool would do if challenged with a spectrum containing multiple structural isomers.

At this point, the nodes in the relationship tree represent masses, not compositions. Accordingly, StrOligo then computes the likely starting compositions for the full glycan (which "generally results in 50 to 100 compositions") and ranks them according to their agreement with the relationship tree. The resulting compositions are then presented to the user, who may select one or more to proceed to the structural analysis phase.

Here StrOligo uses presumed mammalian biosynthetic constraints to greatly reduce the number of structures considered. For example, the only substitution allowed on the reducing-end GlcNAc is a fucose. All structures compatible with both the user-selected composition and the biosynthetic rules are generated, and each structure is then ranked according to its

agreement with the relationship tree. The list of structures is presented to the user in ranked order.

StrOligo ignores cross-ring cleavages and makes no attempt to assign interresidue linkage. Also, because the glycans are unmethylated, the algorithm cannot draw sharp conclusions about the location of residues within the glycan, leaving considerable ambiguity in the relationship tree.

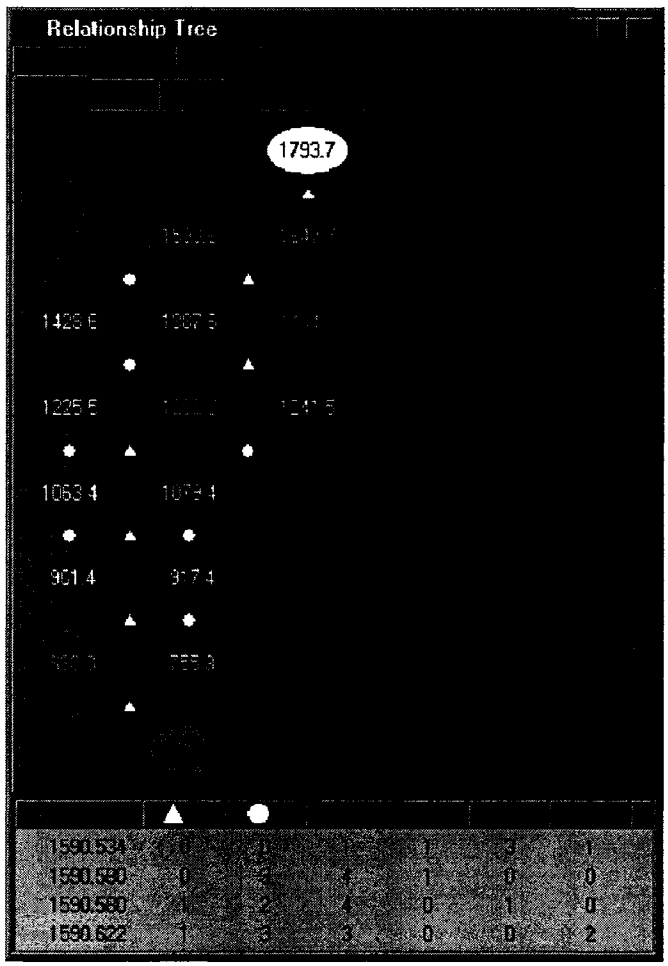


Figure 18: A sample relationship tree as computed by StrOligo. The text box shows some of the possible compositions for ion *m/z* 1590.6.

The algorithm's use of biosynthetic pathways may exclude many actual structures, as our understanding of the pathways will surely increase with time. Additionally, the fact that disease states often present unconventional glycosylation profiles suggests that the use of presumed biosynthetic pathways may hinder the search for biomarkers. The mammalian biosynthetic restrictions also reduce the applicability of the tool to other organisms. For example, the *C. elegans* glycan *m/z* 1928 from (Lapadula 54) includes a Hex-Fuc substitution on the reducing-end GlcNAc, a motif that is disallowed by StrOligo. The authors seem aware of this limitation and state that "efforts are currently being made to include other biosynthetic rules." In a follow-up paper (Ethier 26), the authors report that some biosynthetic constraints had to be removed to process glycans from IgG, beta interferon, and fetuin. The *de novo* approach used by OSCAR does not suffer from these limitations, but the algorithm must correspondingly work harder to avoid being overrun by a multitude of possible structures.

4.6. GLYCH: GLYcan CHaracterization

GLYCH (Tang 81) performs *de novo* interpretations of high energy CID MS^2 spectra of permethylated glycans. The algorithm relies on cross-ring cleavages to make appropriate assignments.

GLYCH defines the Prefix Residue Mass (PRM) m_i as the total mass for the residues of the subtree rooted at residue i. See Figure 19, which is taken from (Tang 81). It also defines the Prefix Residue Feature (PRF) for residue i as (m_i, r_i, b_i), where m_i is the PRM, r_i is the residue type, and b_i is the linkage position from residue i's parent. Therefore any glycan with n residues can be described by the series of PRFs $(m_1, r_1, b_1) \ldots (m_n, r_n, b_n)$.

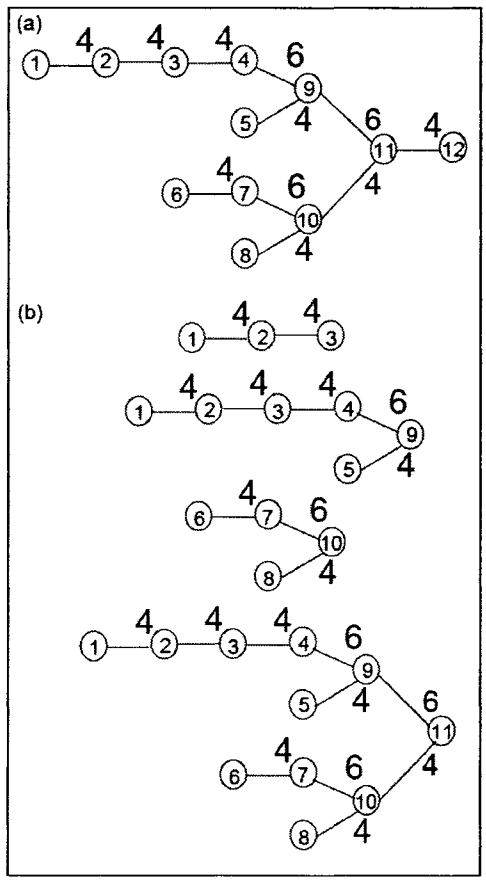


Figure 19: (a) A sample oligosaccharide containing 12 residues. (b) Subtrees rooted by residues r_3, r_9, r_{10}, and r_{11}.

Each theoretically-possible PRF is assigned a score equal to the number of peaks consistent with it. Then a pathway PRF_1 through PRF_n is sought that maximizes the sum of the selected PRFs' scores; this sum is assigned to the structure defined by that pathway.

A post-processing step is applied, where each proposed structure is fragmented *in silico*, and the resulting theoretical spectrum is compared with the experimental. The number of common peaks is used to rank the structures, and the results are presented to the user.

GLYCH removes low intensity peaks before processing the spectrum. It is unclear why this is done (other than improved performance), as these peaks might be the very ones to lend crucial support to the correct glycan structure. Cross-ring cleavages, after all, are usually much lower in intensity than glycosidic cleavages.

Also, the authors leave unanswered the question of what GLYCH does when support for PRF_k is missing because the corresponding low-intensity peak was removed. Now the glycan is actually defined by two sequences: $PRF_1..PRF_{k-1}$ and $PRF_{k+1}..PRF_n$. Can the algorithm typically overcome this gap?

The authors report that GLYCH's scoring algorithm seems to prefer linear structures to branching, and that as the complexity of the glycan increased, so did the number of optimal solutions. The reasons for these behaviors are left unexplained, but the observations cast some doubt on the applicability of this method to the *de novo* assignment of larger, branching structures. In fact, GLYCH supports only binary branching, and so the common bisecting HexNAc motif is unsupported. It is also unclear what GLYCH would do if presented with an MS^2 spectrum of an isomeric mixture.

Even though the glycans analyzed were permethylated, it does not appear that GLYCH takes especially good advantage of the cleavage indications left behind on fragments. Perhaps this is because these cleavages provide more information when they appear in an MS^n disassembly pathway. For comparison, OSCAR relies heavily on these clues.

4.7. The Catalog-Library Method

Tseng's Catalog-Library approach (Tseng 82; Tseng 83) is not a tool, but could readily be automated. As such, it merits discussion here. This method observes that a given substructure, even when embedded in different structures, will fragment to provide a reliable mass spectral fingerprint motif.

To this end, the authors constructed a catalog of five different motifs, gathered from MS^n analysis of glycans that had been fully characterized by NMR. Then experimental spectra for unknown glycans were manually scanned for motifs that appeared in the catalog. Figure 20(a)

shows one motif from the catalog, while (b) shows the same motif appearing at a reduced intensity as part of the spectrum for a larger structure. The authors therefore conclude that the cataloged substructure must be a component of the unknown. Multiple motifs appearing in the same experimental spectrum can identify potentially overlapping substructures, bringing some structural clarity to the overall glycan.

The catalog described contained only five entries and the authors write (Tseng 83) that "a new biological source may, however, require a new catalog based on a different group of substructural motifs." It is unclear why a single "master catalog" could not be constructed and applied to various biological sources.

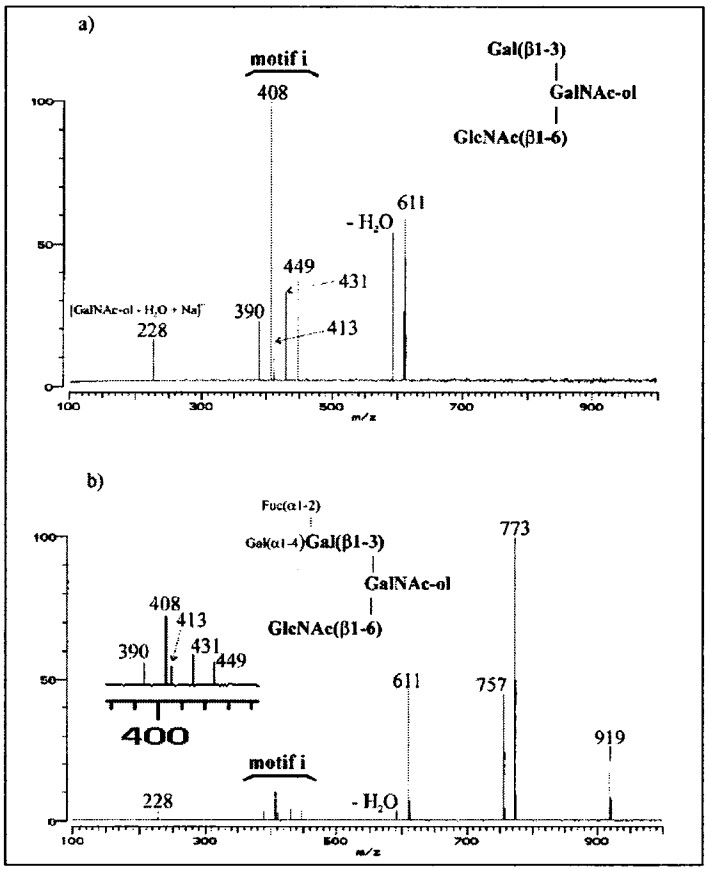


Figure 20: (a) A motif derived from a well-characterized fragment. (b) The same motif as it appears as part of a larger structure.

4.8. STAT: Saccharide Topology Analysis Tool

STAT (Gaucher 31; Leavell 55) is a Web-based tool that attempts to sequence glycans of up to ten residues. STAT accepts a non-hierarchical list of ions selected from multiple MS^n spectra, but does not derive information from the generative relationship between precursor and product ions. Operators manually select between multiple compositions for ambiguous ions. STAT then computes every possible branching topology for the selected starting composition. Next, each ion is considered as a subtree, and topologies that do not embed all subtrees are eliminated from consideration. The remaining structures are scored, with a penalty levied for each cleavage required to extract the fragment from the full glycan. The structures are then sorted by score and presented to the user.

When processing *N*-glycans, STAT excludes certain structures to reduce the candidate set. For example, bisecting HexNAcs are disallowed, as are substitutions on the reducing-end HexNAc. These restrictions would exclude many reported structures.

Because STAT computes all possible topologies for a given composition, it does not scale well with glycan size. For example, it analyzes glycans with eight residues "nearly instantaneously," nine residues "in 1 minute," and 10 residues "in ~5-10 min," with execution time increasing "exponentially" from there. The underlying issue is that "the number of tree structures generated and requiring evaluation increases exponentially." For comparison, OSCAR does not need to generate structures to eliminate them (Lapadula 52).

STAT uses the penalty scoring system because it operates on native glycans. Here, the tool cannot directly observe the number of cleavages required to liberate a fragment from its precursor structure. Because it accepts permethylated glycans, OSCAR extracts cleavage counts directly from the observed masses, eliminating many candidate structures that would have

merely been penalized by STAT. As a simple example for permethylated glycans, a terminal hexose residue can be distinguished from an internal hexose. With native glycans, the two residues have the same mass and cannot be differentiated.

STAT does not support cross-ring fragment ions, whereas OSCAR accepts and processes a wide variety of these to propose interresidue linkages.

4.9. Cartoonist

Cartoonist (Goldberg 32) is a high-performance tool that automatically annotates MS profile scans of *N*-linked permethylated glycans, attaching putative topologies to observed peaks. See Figure 21, taken from (Goldberg 32).

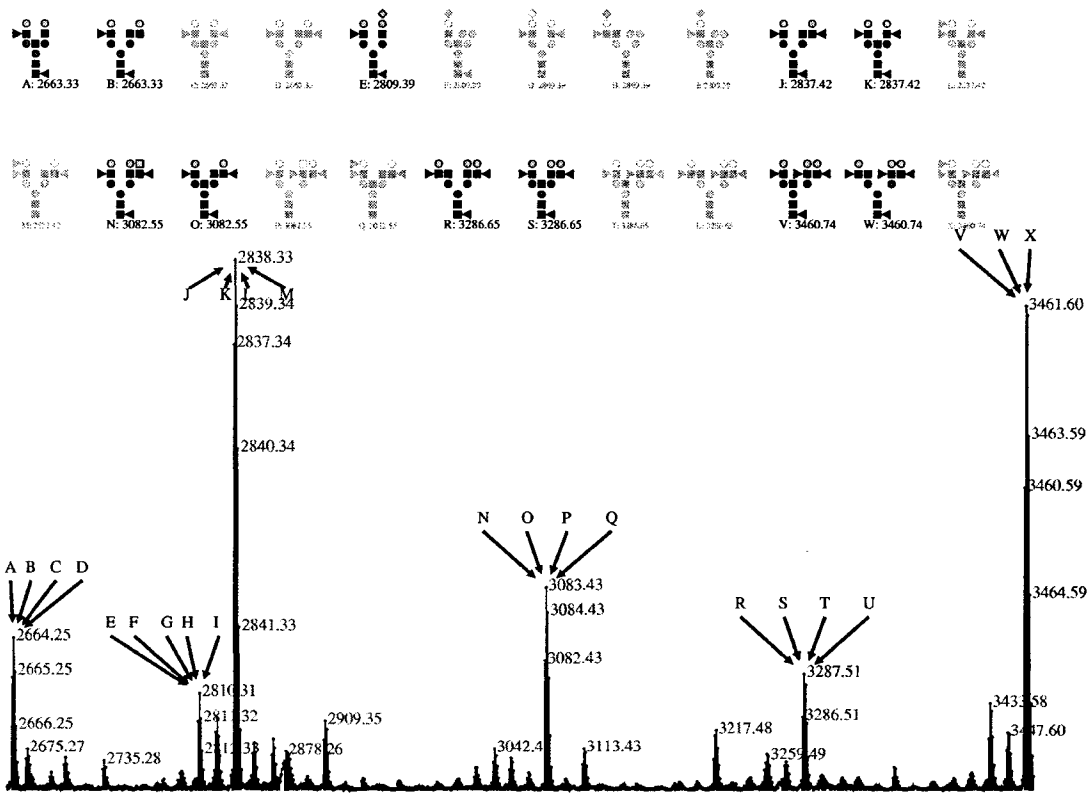


Figure 21: Sample Cartoonist output for portion of a mouse kidney profile spectrum. All matching cartoons are shown, with those of lower rank deemphasized.

Cartoonist first creates a database of potential glycans, along with their masses and predicted isotopic envelopes. Next, from a manually-determined set of 300 reference *N*-glycans, Cartoonist applies a series of biologically-derived rules to create a set of 2800 "cartoons"—that is, the set of topologies it is capable of assigning.

Next a linear calibration is applied to the spectrum by measuring errors from high-confidence peaks. This allows the precisely calibrated spectrum to be used when making composition assignments.

Finally, the tool examines the spectrum. For each isotopic envelope, every cartoon whose mass matches the envelope's putative glycan peak is given a confidence score based upon the match between the experimental and theoretical envelopes. Cartoons with "uncommon" features have their scores penalized. The highest-scoring cartoon is used to annotate the peak, although lower scoring cartoons are available for examination. Impressively, the tool can annotate a typical spectrum in several seconds.

Cartoonist is capable of discovering low-intensity peaks that may represent glycans. However, the lack of MS^2 fragmentation casts some doubt on how often the composition and topology results are correct. Cartoonist does not assign interresidue linkages. Further, it is uncertain how the presence of isomers might affect the program's output.

As with StrOligo, the biosynthetic restrictions used may reduce the applicability of the tool in certain contexts. Again we see that the *m/z* 1928 *C. elegans* glycan from (Lapadula 54) would be excluded from consideration, as would several other structures reported in this document. The authors admirably mention this limitation and state that "these constraints can be removed or altered."

CHAPTER 5:

OSCAR

5.1. Overview

GlySpy's OSCAR algorithm offers *de novo* sequencing capabilities: it accepts a set of MS^n fragmentation pathways as input and reports all possible glycans that are consistent with those data. In this chapter, we present a continuing example of how OSCAR generates structures compatible with the disassembly pathway *m/z* 1187.6 → 894.4 → 649.2 → 431.1 → 259.0.

5.1.1. Deriving Composition and Topology from MS^n Data

To assign the topology of a glycan, OSCAR accepts ion *m/z* values ordered as fragmentation pathways. Using its built-in mass/composition database (Section 3.5), it maps the *m/z* values to plausible fragment compositions, which are then used to infer possible topologies of the original glycan. As described in Section 2.6.3, because the glycans are methylated, their fragments preserve hints of their original linkage. OSCAR uses these cleavages and the constraints imposed by the tree-like structure of glycans to compute the branching of the original structure.

5.1.2. A Detailed OSCAR Example

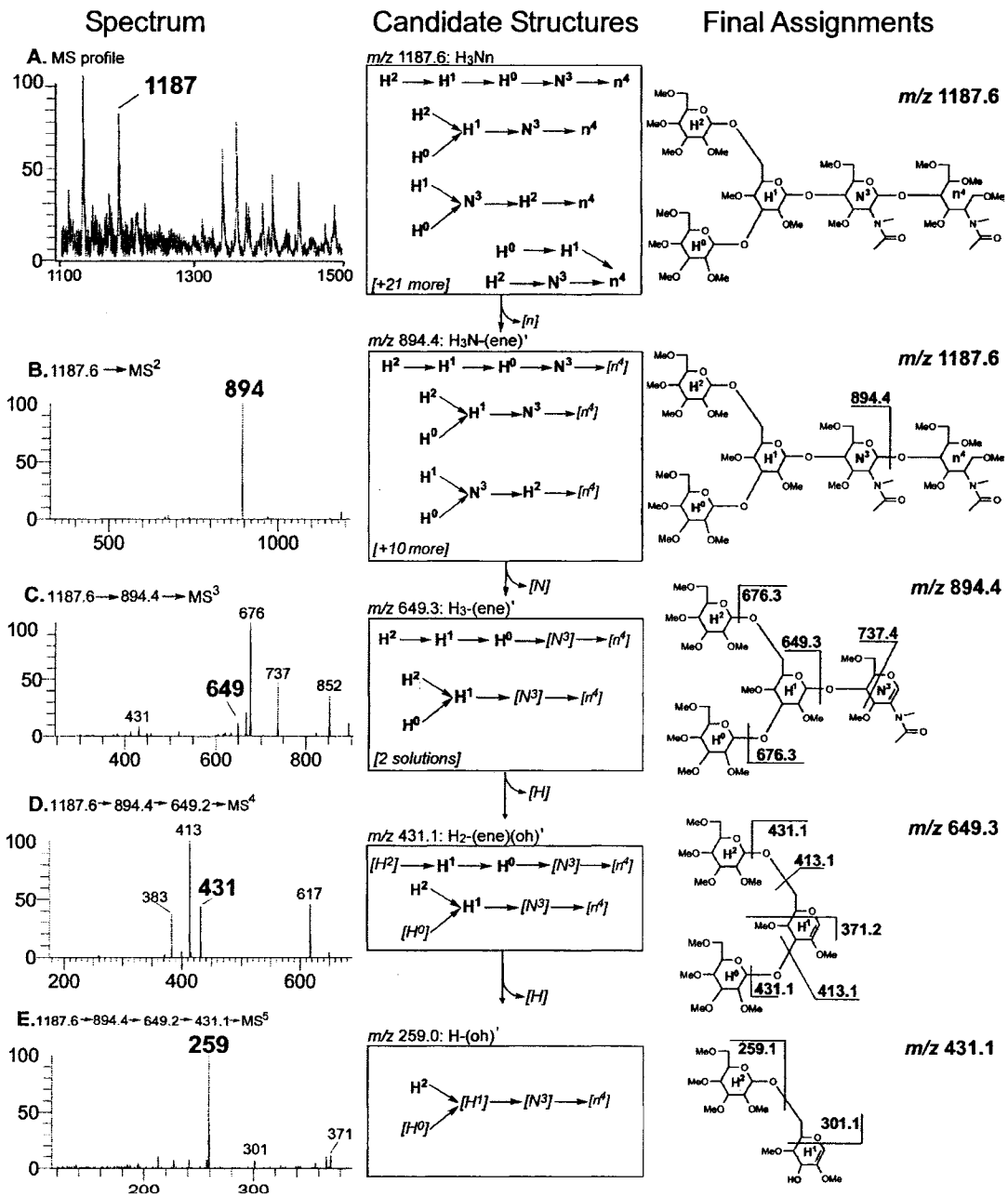


Figure 22: MS^n disassembly of the simplest N-glycan along the pathway *m/z* 1187.6 → 894.4 → 649.2 → 431.1 → 259.0.

Figure 22, adapted from (Lapadula 54), illustrates how OSCAR utilizes the masses and inferred compositions of an MS^n fragmentation pathway to produce a diminishing set of candidate structures (or, in this case, a single structure). The pathway processed here is *m/z*

1187.6 → 894.4 → 649.2 → 431.1 → 259.0 and the structure is the simplest *N*-glycan, namely the H_3Nn trimannosyl core. The left-hand column shows the spectrum for each step. The center column displays the structures consistent with all ions processed to that point: bold type substructures match the product ion composition introduced at each step and lost residues are shown italicized in brackets. By the end of the pathway, only a single structure remains. The right-hand column contains the final assignments of selected ions.

This example is of course a very simple one, containing a single input pathway and a single output structure, neglecting OSCAR's ability to combine multiple pathways and, when necessary, to output multiple structures. However, this example was carefully chosen and forms the basis for an ongoing discussion of OSCAR's implementation.

5.1.3. Invoking OSCAR via GlySpy Commands

The above analysis is accomplished through application of the commands **`AddPathway`** and **`Summarize`**. These commands expose OSCAR's core analytical capabilities and are described, along with other important commands, in Section 5.2. Sections 5.3 and 5.4 cover OSCAR's data structures and the algorithm itself, respectively. Results and discussion then follow.

5.2. Commands and Options

Recall that OSCAR is one component of the GlySpy command-line executable. In this section we discuss a few common OSCAR commands. This will give a sense of how the analyst can use OSCAR to speed structural analysis.

5.2.1. The LabelPathway Command

Compositions for the ions in a pathway are provided by the LabelPathway command, which has this format:

```
LabelPathway [NoCrossRing] [DoNotOptimize] pathway
```

A sample input is shown in Listing 1; the three invocations of LabelPathway are labeled (A), (B), and (C). The command's optional parameters are defined in Table 5.

```
; Specify that the reducing-end residue must be a reduced HexNAc
-ReducingEndResidue n

; A) First show all possible compositions by specifying DoNotOptimize
LabelPathway DoNotOptimize 1187.6_894.4_649.2_431.1_259.0

; B) Now optimize the pathways and throw away impossible compositions
LabelPathway 1187.6_894.4_649.2_431.1_259.0

; C) Further optimize by excluding cross-ring compositions
LabelPathway NoCrossRing 1187.6_894.4_649.2_431.1_259.0
```

Listing 1: Three examples of the LabelPathway command using (A) the DoNotOptimize option, (B) no options, and (C) the NoCrossRing option.

Table 5: LabelPathway options.

Option	Meaning
NoCrossRing	Disallow compositions that include a cross-ring fragment. (Results in glycosidic fragments only.)
DoNotOptimize	Do not apply logical constraints to exclude impossible product/precursor composition combinations.

Listing 1A (`LabelPathway DoNotOptimize ...`) results in an enormous list of possible compositions for each ion in the pathway. Table 6 shows a small selection of these ions. Highlighted in the right column are counts of the many ions excluded from the table. For example, the ion *m/z* 894.4 matches 78 possible compositions with masses within the default ±0.5 Da error window—the four shown plus 74 more, many of which involved cross-ring cleavages. This simple five-ion pathway has returned a total of 163 possible compositions!

Table 6: Selected compositions returned by the `LabelPathway DoNotOptimize` command for each ion in the pathway 1187.6 → 894.4 → 649.2 → 431.1 → 259.0.

Ion *m/z*	Theoretical *m/z* for Selected Returned Compositions
1187.6	1187.61 H_3Nn
894.4	894.38 NS_2-(ene)$_4$(oh)' 894.39 HF_2S-(ene)(oh)$_4$' 894.43 H_3N-(ene)' 894.44 NSn-(oh)$_3$ **...and 74 more**
649.2	649.26 S_2-(ene)$_4$(oh)' 649.30 H_3-(ene)' 649.32 Sn-(oh)$_3$ **...and 44 more**
431.1	431.18 N_2-(ene)$_4$' 431.19 H_2-(ene)(oh)' **...and 24 more**
259.0	259.12 H-(oh)' **...and 10 more**

Many of these compositions are logically inconsistent. For example, there is no possibility that a precursor ion H_3Nn (*m/z* 1187.61) could yield a product ion NS_2-(ene)$_4$(oh)' (*m/z* 898.38). The precursor has no S residues but the product has two! Further, many of the 163 compositions represent exotic cross-ring cleavages that have the correct mass, but which could not have arisen from any listed precursor. (For example, ion *m/z* 259.0 returns 11 compositions, ten of which are cross-ring cleavages that can be ruled out by context.) Clearly the results of Table 6 need to be pruned for these precursor/product relationships. OSCAR scans the composition list multiple times, removing any product composition that could not conceivably come from any of its putative precursor compositions. Likewise, a precursor composition is removed if it could not generate any of the product compositions listed. This sifting is repeated

until the composition list converges. This is the default behavior of the **LabelPathway** command—that is, when the user does not specify the **DoNotOptimize** flag. (In practice, the analyst would *never* specify the **DoNotOptimize** flag. It is provided solely to illustrate the importance of pruning according to precursor/product relationships.) With this optimization enabled, Listing 1B produces the unabridged composition list of Table 7—a total of only eight compositions, down from 163. Clearly this optimization is an important step in controlling the complexity of composition lists.

Table 7: All compositions returned by the LabelPathway command for each ion in the pathway *m/z* 1187.6 → 894.4 → 649.2 → 431.1 → 259.0.

Ion *m/z*	Theoretical *m/z* for All Returned Compositions
1187.6	1187.61 H_3Nn
894.4	894.43 H_3N-(ene)' 894.45 H_2N-^{1,5}A[n]'
649.2	649.30 H_3-(ene)' 649.33 H_2-^{1,5}A[n]'
431.1	431.19 H_2-(ene)(oh)' 431.21 H-(oh)^{1,5}A[n]'
259.0	259.12 H-(oh)'

Listing 1C shows the **LabelPathway** command used with the **NoCrossRing** option given. As you would expect, this causes cross-ring compositions to be excluded from consideration. The corresponding output is summarized in Table 8. Now only one possible composition is listed for each ion *m/z*.

The analyst uses the **NoCrossRing** option when convinced that the observed fragments can be explained by glycosidic cleavages, as is usually true of high-abundance fragments. (See

Section 2.6.3 on page 13.) Requiring the use of this option follows GlySpy's guiding design principle that all possibilities should be considered unless otherwise instructed by the analyst.

Table 8: All compositions returned by the `LabelPathway NoCrossRing` command for each ion in the pathway 1187.6 → 894.4 → 649.2 → 431.1 → 259.0.

Ion *m/z*	Theoretical *m/z* for All Returned Compositions
1187.6	1187.61 H_3Nn
894.4	894.43 H_3N-(ene)'
649.2	649.30 H_3-(ene)'
431.1	431.19 H_2-(ene)(oh)'
259.0	259.12 H-(oh)'

The LabelPathway command is useful as a stand-alone command, allowing the analyst to quickly ascertain likely compositions for prospective pathways. However, the same composition list optimization described here is crucial for the next command, **`AddPathway`**.

5.2.2. The AddPathway and Summarize Commands

As shown in Figure 12 on page 19, OSCAR accepts one or more disassembly pathways and produces a set of glycan structures that are consistent with all of the constraints implied by those pathways. The process occurs in two steps. First the analyst issues one or more **`AddPathway`** commands and, second, executes a **`Summarize`** command to generate the structures. This is demonstrated in Listing 2. When executed, this input generates a single topology, H(H)HNn, which is the expected *N*-linked core.

```
AddPathway 1187.6_894.4_649.2_431.1_259.0
Summarize
```

Listing 2: A simple demonstration of the AddPathway and Summarize commands.

5.2.3. The AddSpectrumFile Command

Several GlySpy commands operate on entire spectra instead of analyst-selected pathways. Currently, GlySpy supports the ".raw" file format produced by Thermo Fisher LTQ ion trap. To add a spectrum file for processing, the analyst uses the command **`AddSpectrumFile filename`**.

5.2.4. The LabelSpectra Command

One useful command that processes spectra added by the AddSpectrumFile command is LabelSpectra. This command scans each added spectrum and reports the plausible disassembly pathways found, along with composition assignments for every ion in every pathway. The command has the format:

```
LabelSpectra [NoCrossRing] [NLinked] MZ-target rel-intensity
```

The **`MZ-target`** parameter gives the *m/z* of the unfragmented glycan. The **`rel-intensity`** parameter specifies a relative intensity cutoff; peaks which fall below this threshold are ignored. The **`NoCrossRing`** and **`NLinked`** options restrict assigned compositions in the obvious ways. The command also supports an **`EstimateTopologies`** option, but discussion of this option is beyond the scope of this chapter.

```
-ReducingEndResidue n
AddSpectrumFile OVA_1187_894_676.raw
LabelSpectra NoCrossRing 1187.61 2
```

Listing 3: Sample input demonstrating the AddSpectrumFile and LabelSpectra commands.

Listing 3 shows a simple use of the AddSpectrumFile and LabelSpectra commands. The spectrum added is shown as Spectrum A-39 on page 229 in the appendix. The output of the LabelSpectra command identifies six pathways terminating on this spectrum that have non-cross-ring composition interpretations. The interpretation of the first three ions in each pathway is identical, and shown in Table 9.

Table 9: Composition assignments for the pathway prefix *m/z* 1187.6 → 894.4 → 676.3.

Ion *m/z*	Composition
1187.61	H_3Nn
894.43	H_3N-(ene)'
676.32	H_2N-(ene)(oh)'

Table 10: Composition assignments for the six terminal glycosidic ions found on Spectrum A-39.

Terminal Ion *m/z*	Composition
241.11	H-(ene)'
259.12	H-(oh)'
268.12	N-(ene)(oh)'
431.19	H_2-(ene)(oh)'
449.20	H_2-$(oh)_2$'
458.20	HN-(ene)$(oh)_2$'

The terminating ions for each of the six pathways are shown in Table 10. Several ions on Spectrum A-39 are above the 2% relative intensity cut-off, but are not output by the command. This is because these ions have no possible glycosidic interpretation. For example, ion *m/z* 519.3 represents a cross-ring fragment.

5.3. Data Structures

OSCAR's main data structures are *fork*, *solution*, *mono*, and *box*, which can be understood as follows:

- A **fork** is *one interpretation* of the input MS^n ions, and can produce a set of glycans that are consistent with that interpretation. Since each ion can map to multiple compositions and/or specific residues, a separate fork is created for every combination of these mappings[4].
- A **solution** contains a set of forks that covers *all possible interpretations* of the input ions. The union of the glycans produced by all contained forks represents all possible glycan structures given the selected ions.
- A **mono** is a single monosaccharide residue.
- A **box** encloses a set of monos that are known to belong to a single MS^n ion.

Roughly speaking, a fork represents one "slice" of the entire problem, which is itself represented by the solution. Each fork contains a set of monos, along with a series of boxes that encloses subsets of those monos. Each box groups together monos that are assumed, in this fork, to form a connected subtree that is embedded in the final glycan structure. (Referring to the equivalent terms defined in Table 1 on page 10 may be useful.)

Figure 23 illustrates the relationships between solutions, forks, monos and boxes.

---

[4] The term "fork" comes from a quote attributed to Yogi Berra: "When you come to a fork in the road, take it." This is exactly the search strategy used by OSCAR: when multiple interpretations are possible, all of them are explored, one per fork.

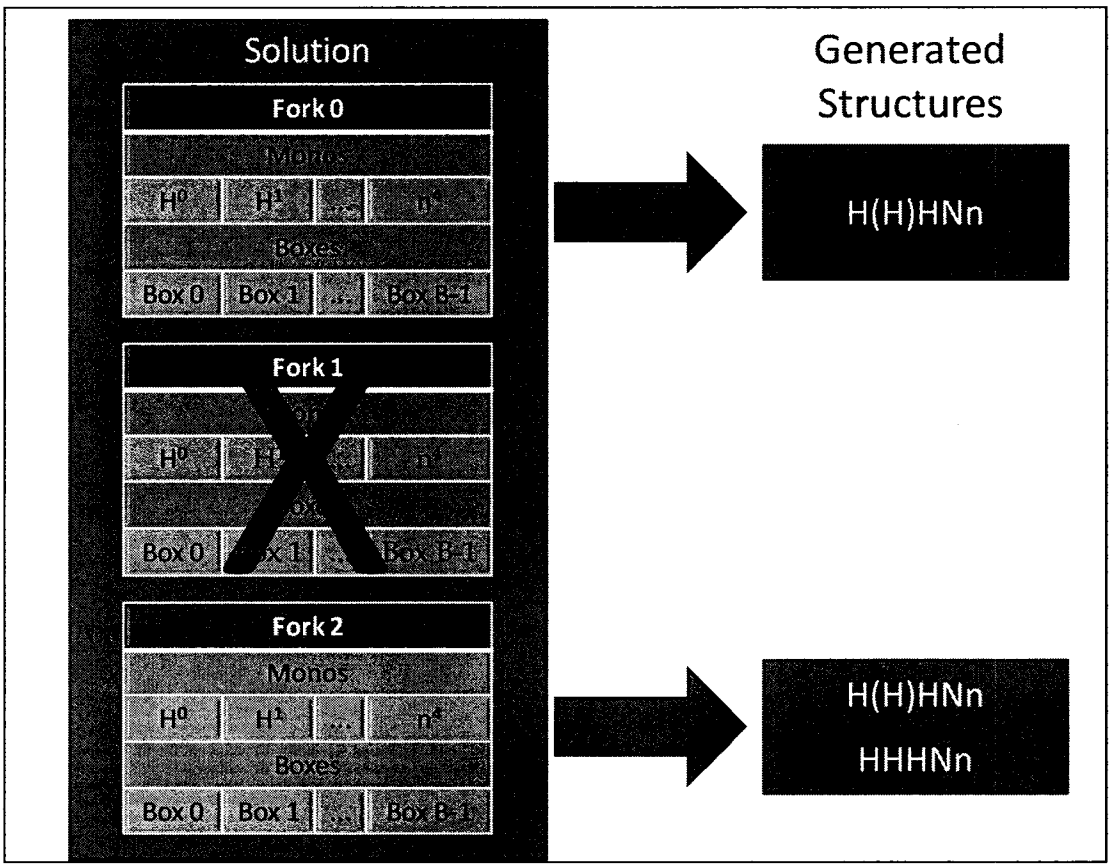


Figure 23: A solution containing three forks, numbered 0..2. Each fork contains a set of monos and boxes. Fork 1 has been marked as dead because some internal inconsistency was discovered. Forks 0 and 2 are still alive and can generate glycan topologies when requested.

5.3.1. Fork

A fork is one interpretation of the input MS^n ions, and can produce a set of glycans that are consistent with that interpretation. (Consistent glycans are those that are not refuted by any of the input MS^n data. For example, if the input MS^n data specify that the only F in a glycan is definitely connected to n, then the solution will only generate glycans where F is connected to n.) As more ions are selected and added to the fork, the fork becomes more specific, generating fewer and fewer consistent glycans. When a fork is found to generate no consistent glycans, it is marked as "dead," as demonstrated by fork 1 in Figure 23.

A fork contains:

- A set of monos
- A list of boxes that describe how the various MS^n inputs have been mapped to those monos
- A score that indicates how constrained the fork is, with a *lower* score representing a more constrained and therefore more specific fork. (The score is roughly analogous to the degrees of freedom remaining within a fork and is described in Section 5.4.3.4 on page 76.)

Glycan structures generated by the forks may overlap. In Figure 23, forks 0 and 2 both generate the branched *N*-linked core motif H(H)HNn. Fork 2 also produces a linear HHHNn structure. Before the results are presented to the analyst, these three structures are collected and the duplicate is eliminated.

5.3.2. Solution

A Solution is simply a set of forks. The solution as a whole can generate the entire set of consistent glycans. As new MS^n ion fragments are added to GlySpy, the solution can grow or shrink. Forks are added when multiple search paths must be explored, and are removed when they are discovered to be internally inconsistent or redundant with another fork in the solution.

5.3.3. Mono

A mono represents a monosaccharide residue and contains:

- The type of mono (H, F, N, S, h, f, or n)
- The index of the mono (running from 0 to N-1, if there are N residues in the glycan)

- A set representing the monos that might *possibly* be this mono's parent in the glycan (called **`ParentPossible`**)
- A set that represents the mono that is *definitely* this mono's parent (**`ParentDefinite`**)
- A set representing the monos that might *possibly* be this mono's children in the glycan (**`ChildrenPossible`**)
- A set that represents the monos that are *definitely* this mono's children (**`ChildrenDefinite`**)
- A set containing the number of possible children this mono might have (**`NumChildrenPossible`**)
- A set containing the possible linkage positions (2, 3, 4, or 6) between this mono and its parent (**`Linkage`**)

When created, the mono's fields are initialized as follows:

- **`ParentPossible`**: All monos except self
- **`ParentDefinite`**: Empty
- **`ChildrenPossible`**: All monos except self
- **`ChildrenDefinite`**: Empty
- **`NumChildrenPossible`**: { 0, 1, 2, 3, 4 }
- **`Linkage`**: { 2, 3, 4, 6 }

OSCAR restricts **`ParentPossible`**, **`ChildrenPossible`**, **`NumChildrenPossible`**, and **`Linkage`** to fewer and fewer possible values. When the identity of a Mono's parent or child is

learned with certainty, the **ParentDefinite** and **ChildrenDefinite** fields are set. For example, if a Mono has **NumChildrenPossible** = { 1 } (meaning that the given mono must have exactly one child) and **ChildrenPossible** = { H^0 }, then **ChildrenDefinite** can be set to { H^0 }.

Implementation digression: Some inference rules, discussed both below and in APPENDIX B: SAMPLE OSCAR INFERENCE RULES, can draw useful inferences knowing only the *possible* parents and children, whereas other inference rules apply only when the parent and children are *definitely* known. This is why both **Possible** and **Definite** sets are computed. It could also be argued that having both parent and child information available is redundant, since one can be computed from the other. In our experience, some inference rules are much easier to write, understand, and debug given parent information, whereas others are more naturally expressed in terms of children. The minimal additional storage space pays for itself in ease of program maintenance and improved execution time.

5.3.4. Box

A box represents a single ion and maps that ion to a set of monos plus a number of reducing-end and non-reducing-end scars. The monos within the box must form a connected subtree embedded within the glycan. The scars on the box represent the cleavages necessary to extract this subtree from the glycan.

Each box contains:

- A set of monos contained by this box
- The type of the ion's reducing-end scar, and the number and types of its non-reducing-end scars.

- If an MS^n ion maps to H_3-$(oh)_3$', there will be one corresponding box that contains three H monos and has two non-reducing-end (oh) scars and one reducing-end (oh) scar.

- A unique ordinal index, with the first box in a fork always numbered zero (called **Index**)
- A link to a complementary box, if any (more on complementary boxes later)
- A set that represents the monos that might *possibly* be the root of this box (**RootPossible**)
- A set that represents the mono that is *definitely* the root of this box (**RootDefinite**)
- A set that represents the mono that this box's root might *possibly* connect to (**RootParentPossible**)
- A set that represents the mono that this box's root *definitely* connects to (**RootParentDefinite**)
- A set containing the possible linkage positions (2, 3, 4, or 6) between this box and its parent (**Linkage**)

As with mono, box maintains **Possible** and **Definite** variations of both **Root** and **RootParent**. Again, this is because some inference rules can be applied knowing only which roots and root parents are possible, but other inference rules are valid only when the root and root parent are known with certainty.

When created, the box's fields are initialized as follows:

- **RootPossible**: All monos contained by this box

- **RootDefinite**: Empty
- **RootParentPossible**: Usually, any mono not contained by this box. However, if the box has no parent scar, then the subtree defined by this box must include the root of the entire glycan, and **RootParentPossible** is initialized to empty.
- **RootParentDefinite**: Empty
- **Linkage**: { 2, 3, 4, 6 }

Over time, OSCAR attempts to restrict **RootPossible**, **RootParentPossible**, and **Linkage** to fewer and fewer possible values. When the identity of a box's root or root parent is learned with certainty, the **RootDefinite** and **RootParentDefinite** fields are set.

5.4. Algorithm

The Oligosaccharide Subtree Constraint Algorithm is the computational heart of GlySpy. It accepts MS^n input pathways, adds/removes forks to/from the solution, and applies logical constraints to reduce the number of glycans generated by each fork.

We will describe the algorithm largely by way of example, showing how OSCAR derives the branching topology of the *N*-linked core (Figure 24) by processing the single input pathway *m/z* 1187.6 → 894.4 → 649.2 → 431.1 → 259.0 (Listing 4).

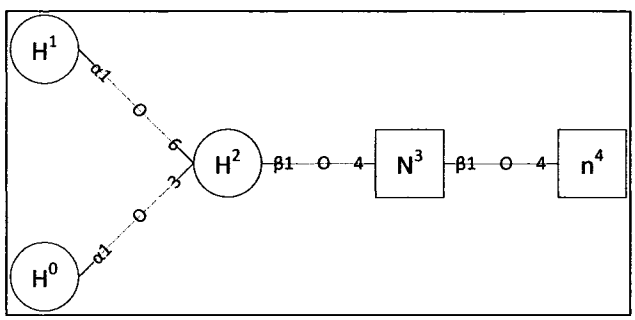


Figure 24: The five-residue *N*-linked core.

```
-ReducingEndResidue n
AddPathway 1187.6_894.4_649.2_431.1_259.0
Summarize
```

Listing 4: The input used as an example to illustrate the operation of OSCAR.

A complete description of even this simple example is beyond the scope of this (or any reasonable) document, but we strive to present enough detail to illustrate the key data structures and algorithmic steps.

5.4.1. Overview

OSCAR maintains a single solution that contains a set of forks, each of which can generate a number of consistent glycans. Each MS^n input pathway selected by the user is mapped to a number of entries in the composition database; each of these compositions is then applied to the existing forks. Existing forks may need to be copied ("forked") before this application can be done.

Forks are tentative hypotheses that are discarded if discovered to contain internal contradictions. They are small data structures that are easy to copy and discard, and are created whenever multiple interpretations of an input ion must be examined.

Logical inference rules are then applied iteratively to each fork. Forks that can generate no consistent glycans are marked as dead and removed from the solution. Also, isomorphic (redundant) forks are removed from the solution, to ensure that the solution does not grow to include an exponential number of isomorphic forks.

It is important to note that OSCAR is *de novo* and utilizes no knowledge of previously-reported glycans or presumed biosynthetic pathways. Almost all of the constraints applied are straightforward corollaries that arise from the tree structure of glycans; a few remaining

constraints are command-line options available to the operator (such as whether the glycan search should be constrained to *N*-linked glycans only).

5.4.2. Boxes, Subtrees and Ions

A glycan is a well-formed tree in the classic computer science sense: every node (mono) in the tree (glycan) has exactly one parent, except for the root of the tree, which has no parent; cycles are not allowed. This means that the initial box, which contains the entire glycan, also, by definition, represents a well-formed tree.

A product ion is also a well-formed subtree. You can visualize a product ion as a subtree that was embedded in the original glycan. The MS^n process used to fragment precursor ions into product ions severs individual parent/child bonds in the glycan, but rarely forms new bonds.

Since each box in a fork represents either the initial glycan or a product ion of the glycan, it follows that ***every box in the fork represents a well-formed subtree***. This observation is key to understanding how OSCAR both makes progress and discards inconsistent forks.

For example, since a box is a subtree and every subtree has exactly one root, we know that exactly one of the monos in the box must be the root of that subtree. If OSCAR at some point deduces that two different monos must be the root of the same box, or that no monos can possibly be the root of the box, then the algorithm has detected an inconsistency. The fork containing that box will be marked as inconsistent and removed from the solution. Consistency checks like these are applied to every box in every fork, and in practice prove very efficient at pruning inconsistent forks.

5.4.3. OSCAR's Main Phases

An outline of how OSCAR's `AddPathway` command processes disassembly pathways is shown in Figure 25.

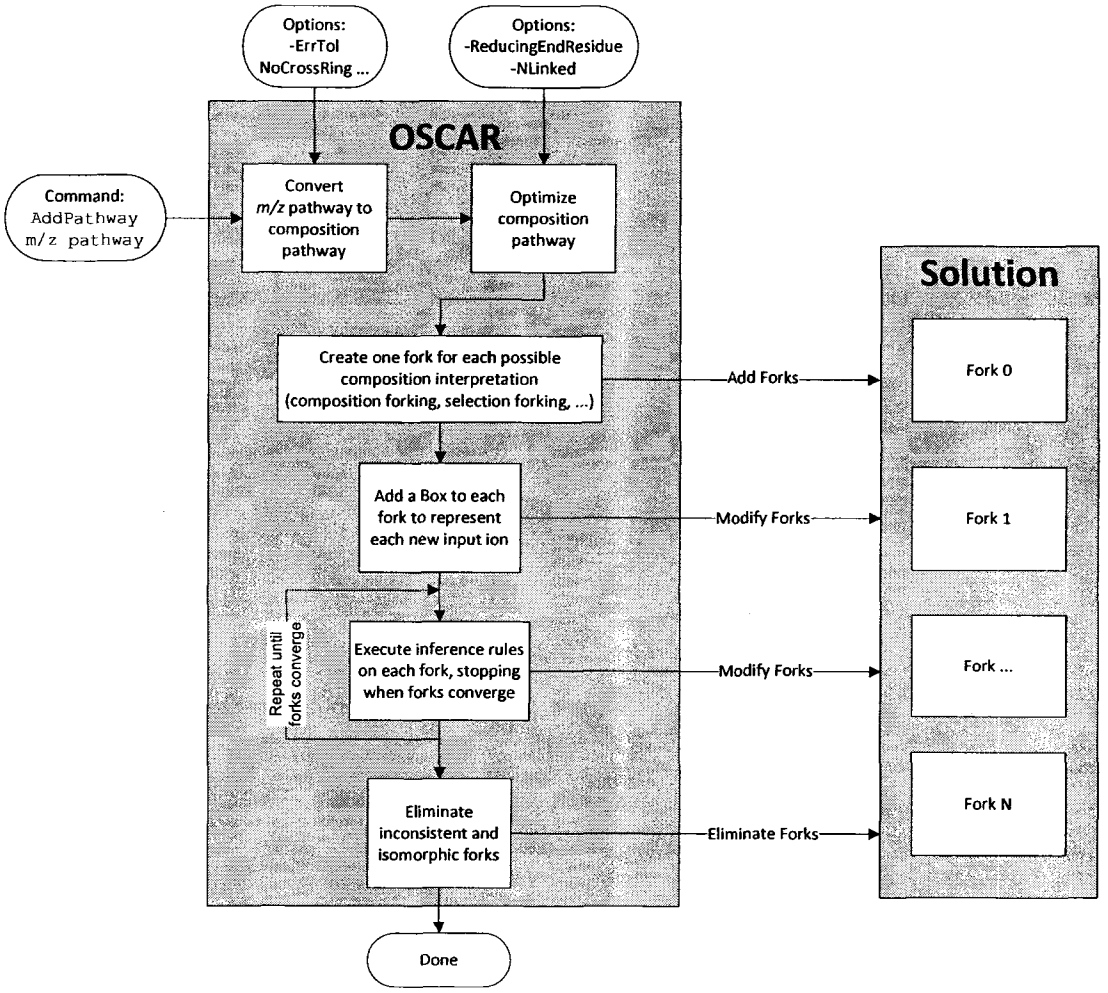


Figure 25: A flowchart representing OSCAR's AddPathway command.

5.4.3.1 Initial State

The solution is set to empty (containing no forks).

5.4.3.2 AddPathway 1187.6_894.4_649.2_431.1_259.0

The `AddPathway 1187.6_894.4_649.2_431.1_259.0` command from Listing 4 defines the observed monoisotopic mass of the glycan of interest as 1187.6. Because the default error tolerance is 0.5 Da, GlySpy retrieves from its database all compositions whose theoretical mass falls within the range *m/z* 1187.6±0.5. Two compositions are found, H_2N_2h and H_3Nn, but because the input listing also includes the option `-ReducingEndResidue n`, only the second is

retained—the first does not contain an n residue. Fork 0 is created for the glycan composition H_3Nn. Within the fork, box 0 is created to represent the ion *m/z* 1187.6, as shown in Table 11. Because of the structure of trees, a mono may not be its own parent or child, and so these possibilities have been excluded, as indicated by the boldface, double-strikeout entries. As this example progresses, these excluded entries will be removed from future fork diagrams, and the boldface, double-strikeout text will be reserved for the latest set of changes in the fork.

Because box 0 has no parent scar, its RootParent is initialized to the empty set. Recall that RootParent represents the mono to which this box's subtree connects. However, as the box has no parent scars, it must contain the root of the glycan and will not connect to any mono—hence, the initial empty value for RootParent.

Table 11: Fork 0 after adding H_3Nn as the glycan's composition.

Fork 0					
H^0	~~H^0~~ H^1 H^2 N^3 n^4		~~H^0~~ H^1 H^2 N^3 n^4		0 1 2 3 4
H^1	H^0 ~~H^1~~ H^2 N^3 n^4		H^0 ~~H^1~~ H^2 N^3 n^4		0 1 2 3 4
H^2	H^0 H^1 ~~H^2~~ N^3 n^4		H^0 H^1 ~~H^2~~ N^3 n^4		0 1 2 3 4
N^3	H^0 H^1 H^2 ~~N^3~~ n^4		H^0 H^1 H^2 ~~N^3~~ n^4		0 1 2 3 4
n^4	H^0 H^1 H^2 N^3 ~~n^4~~		H^0 H^1 H^2 N^3 ~~n^4~~		0 1 2 3 4
0	1187.61	H_3Nn	H^0 H^1 H^2 N^3 n^4	H^0 H^1 H^2 N^3 n^4	~~H^0~~ ~~H^1~~ ~~H^2~~ ~~N^3~~ ~~n^4~~

OSCAR continues to process the pathway *m/z* 1187.6_894.4_649.2_431.1_259.0 one ion at a time. As it converts each ion to its set of possible compositions, those compositions are reduced via the same process as outlined in Table 6 and Table 7 (pages 48 and 49): impossible precursor and product compositions are excluded from further consideration. In the end, only

the compositions of Table 7 remain to be processed. A subset of these compositions end up gathered in fork 0 as shown in Table 12; additional forks, not shown, are created to address other possibilities.

Table 12: Fork 0 after adding an additional Box for each ion in the pathway. This is the initial state to which inference rules will be applied.

Fork 0					
H^0	$H^1 H^2 N^3 n^4$		$H^1 H^2 N^3 n^4$		0 1 2 3 4
H^1	H^0 $H^2 N^3 n^4$		H^0 $H^2 N^3 n^4$		0 1 2 3 4
H^2	$H^0 H^1$ $N^3 n^4$		$H^0 H^1$ $N^3 n^4$		0 1 2 3 4
N^3	$H^0 H^1 H^2$ n^4		$H^0 H^1 H^2$ n^4		0 1 2 3 4
n^4	$H^0 H^1 H^2 N^3$		$H^0 H^1 H^2 N^3$		0 1 2 3 4
0	1187.61	H_3Nn	$H^0 H^1 H^2 N^3 n^4$	$H^0 H^1 H^2 N^3 n^4$	
1	894.43	H_3N-(ene)'	$H^0 H^1 H^2 N^3$	$H^0 H^1 H^2 N^3$	n^4
2		CH:1 PAR:0	n^4	n^4	
3	649.30	H_3-(ene)'	$H^0 H^1 H^2$	$H^0 H^1 H^2$	$N^3 n^4$
4		CH:1 PAR:1	N^3	N^3	$H^0 H^1 H^2$ n^4
5	431.19	H_2-(ene)(oh)'	$H^0 H^1$	$H^0 H^1$	$H^2 N^3 n^4$
6		CH:0 PAR:1	H^2	H^2	$H^0 H^1$ $N^3 n^4$
7	259.12	H-(oh)'	H^0	H^0	$H^1 H^2 N^3 n^4$

Boxes 0, 1, 3, 5, and 7 represent the five ions in the pathway. Each has the expected composition and has been assigned a matching set of monos. For example, ion *m/z* 649.3 has been assigned the composition H_3-(ene)', with the three hexose residues designated as H^0, H^1, and H^2. This means that, in this fork, these three hexose residues must form a connected subtree that can be removed from the glycan with a single reducing-end cleavage.

5.4.3.2.1 Complementary Boxes

The three remaining boxes—2, 4, and 6—are known as ***complementary boxes***, with each box containing a set of residues that form a connected subtree within the glycan. OSCAR can often assign the residues lost during fragmentation to their own box. For example, when ion *m/z* 1187.6 was fragmented to produce *m/z* 894.4, the lost residue (n^4) was placed by itself in box 2. In this particular example, all the complementary boxes contain a single residue, but this is not true in all cases. When a disaccharide or larger fragment is lost, OSCAR places all of the lost residues in their own box if the algorithm can prove that they in fact must form a subtree. In cases where multiple residues may have been lost because of multiple cleavages, no complementary box is created.

Complementary boxes can be identified by their lack of a composition in the table and always follow the box to which they are complementary (e.g., box 2 is complementary to box 1). Even though we do not have a full composition for the complementary boxes, we can calculate the total number of child (non-reducing end) and parent (reducing end) scars on the monos contained by the box. These are listed, respectively, as CH and PAR in the Comp/Scars column.

Complementary boxes are created only when OSCAR can prove that the precursor required *exactly one cleavage* to create the product and the lost residues. The product ion and its complement will therefore have, combined, one more parent scar and one more child scar than their common precursor did. (If the cleavage occurs between monos X and Y in the precursor, then X and Y will end up in different products and either X or Y will have a new parent scar and the other will have a new child scar.)

These relationships can be represented algebraically:

$$ParentScars_{Precursor} = ParentScars_{Product} + ParentScars_{Complement} - 1$$

$$ChildScars_{Precursor} = ChildScars_{Product} + ChildScars_{Complement} - 1$$

Rearranging yields a form that lets us compute the number of parent and child scars on the inferred complementary ion:

$$ParentScars_{Complement} = ParentScars_{Precursor} - ParentScars_{Product} + 1$$

$$ChildScars_{Complement} = ChildScars_{Precursor} - ChildScars_{Product} + 1$$

In Table 12, we are able to calculate that complementary box 2 must have 0 parent scars (0-1+1) and 1 child scar (0-0+1). In this way, OSCAR knows that the monos in box 2 must collectively have no parent scars and one child scar when those monos are cleaved from the glycan. In this case, since box 2 contains the single mono n^4, we see that OSCAR can quickly infer that n^4 must have only a single child. Transferring information from box 2 to mono n^4 illustrates how information can flow freely around the fork. It is precisely this free-form information exchange that makes OSCAR efficient and versatile, but, as we shall see, also makes it very difficult to describe with any brevity.

5.4.3.2.2 Forking

Table 12 shows just one fork, but for the input of Listing 4, a total of seven forks are created. Several ***composition forks*** are created to cover the various cross-ring compositions of Table 7, and several ***selection forks*** are created where different H monos are selected for boxes 5, 6, and 7. In the end, six forks will be marked as dead, five because they are isomorphs of other forks, and the sixth because it is internally inconsistent. The details of these forks are omitted for brevity, and we instead focus on how OSCAR's inference rules drive fork 0 toward the single expected structure. Even here, many details must be omitted.

5.4.3.3 Run Inference Rules

Next, OSCAR executes a series of inference rules on each fork, stopping when each fork's score stabilizes, meaning that no further progress is being made. (Scoring is discussed in Section 5.4.3.4, and a selection of inference rules is detailed in APPENDIX B: SAMPLE OSCAR INFERENCE RULES.) Each inference rule is implemented as a separate C++ function and has one of three types, based on the data structure that the rules wishes to interrogate: (1) a box data structure, (2) a mono data structure, or (3) an fork data structure. These rules, classified as box-centric, mono-centric, and fork-centric, are partitioned and applied by type. See the pseudocode of Listing 5 for details. Note that this architecture makes it straightforward to add a new inference rule: implement the rule as a function and add the function to the list of inference rules.

```
For each fork F in the solution do {
    while F is alive and F's score is decreasing do {
        Apply every box-centric inference rule to each box in fork F
        Apply every mono-centric inference rule to each mono in fork F
        Apply every fork-centric inference rule directly to fork F
    }
}
```

Listing 5: Pseudocode for the application of inference rules to the forks in a solution. Three difference types of rules (box-centric, mono-centric, and fork-centric) are repeatedly applied to each fork until the fork's score stabilizes, signaling that no further progress is being made.

The inference rules modify the various fields of the monos and boxes in a fork, attempting to infer the set of glycans that are consistent with constraints imposed by each mono and box. Each inference rule is written in such a way that it can be applied at any time, in any order; this eliminates the problem of ordering the execution of the inference rules and opens the possibility of a future parallel or distributed implementation.

In all cases, inference rules attempt to use the subtrees defined by a fork's boxes to:

- Shrink the Possible sets (**ParentPossible** and **ChildrenPossible** in mono; **RootPossible** and **RootParentPossible** in box; **MSRootPossible** in fork)
- Grow the Definite sets (**ParentDefinite** and **ChildrenDefinite** in mono; **RootDefinite** and **RootParentDefinite** in box)
- Shrink the **NumChildrenPossible** set in mono
- Shrink the **Linkage** set in both mono and box

Currently, GlySpy implements over 50 inference rules. To illustrate a few of them, we continue to trace the evolution of the fork shown in Table 12. When an interesting inference rule changes the fork, we note what it is doing and why the change is valid. The vast majority of these rules use the properties of trees to make progress; the actual amount of chemistry- or glycan-specific knowledge in these rules is insignificant.

As mentioned above, a complete accounting of the processing of even this single fork is beyond the scope of this document. For this extremely simple example, 55 separate applications of 24 different inference rules were able to constrain the fork during processing[5]. Multiply this by the seven forks created and you may appreciate why the following presentation is heavily abridged.

5.4.3.3.1 Apply Inference Rule ApplyMSRootToAnnBox to Box 0

Because the fork's composition includes a reduced residue (n^4), and because reduced residues must be at the reducing end of the glycan, we know that n^4 is the root of the entire

[5] For this example, many of the 50+ inference rules—for example, those dealing with cross-ring cleavages—contributed nothing to the processing of this fork. They were still executed, but caused no change to the fork. Other rules were successfully applied multiple times.

tree. The rule ApplyMSRootToAnnBox applies this information to each box[6]. If the box contains the root mono, then that mono must also be the root of the subtree defined by the box. (Recall that the monos in a box by definition must form a connected subtree.) This rule affects the Root field of box 0, eliminating H^0, H^1, H^2, and N^3 as possibilities, leaving only n^4. See Table 13.

Because the inference rule ApplyMSRootToAnnBox is applied to a box, we consider this rule to be box-centric. The next few examples will also be box-centric, but a mono-centric rule will be discussed in Section 5.4.3.3.5.

Table 13: Fork 0 after applying the inference rule ApplyMSRootToAnnBox to box 0.

Fork 0					
H^0	$H^1 H^2 N^3 n^4$		$H^1 H^2 N^3 n^4$		0 1 2 3 4
H^1	H^0 $H^2 N^3 n^4$		H^0 $H^2 N^3 n^4$		0 1 2 3 4
H^2	$H^0 H^1$ $N^3 n^4$		$H^0 H^1$ $N^3 n^4$		0 1 2 3 4
N^3	$H^0 H^1 H^2$ n^4		$H^0 H^1 H^2$ n^4		0 1 2 3 4
n^4	$H^0 H^1 H^2 N^3$		$H^0 H^1 H^2 N^3$		0 1 2 3 4
0	1187.61	H_3Nn	$H^0 H^1 H^2 N^3 n^4$	~~$H^0 H^1 H^2 N^3$~~ n^4	
1	894.43	H_3N-(ene)'	$H^0 H^1 H^2 N^3$	$H^0 H^1 H^2 N^3$	n^4
		CH:1 PAR:0	n^4	n^4	
3	649.30	H_3-(ene)'	$H^0 H^1 H^2$	$H^0 H^1 H^2$	$N^3 n^4$
4		CH:1 PAR:1	N^3	N^3	$H^0 H^1 H^2$ n^4
5	431.19	H_2-(ene)(oh)'	$H^0 H^1$	$H^0 H^1$	$H^2 N^3 n^4$
6		CH:0 PAR:1	H^2	H^2	$H^0 H^1$ $N^3 n^4$
7	259.12	H-(oh)'	H^0	H^0	$H^1 H^2 N^3 n^4$

[6] Rules are named according to their implementing C++ functions, hence the slight awkwardness.

5.4.3.3.2 Apply Inference Rule InferNumChildrenForSingleton to Box 2

Next, the inference rule InferNumChildrenForSingleton is successfully applied to box 2. This rules states that if a box has N child scars and contains a single mono, then each of those child scars must belong to that mono. In this case, the rule applies to box 2 and its single mono, n^4. Because box 2 has a single child scar (as shown in the Comp/Scars column of Table 14), n^4 must also have a single child scar. This is reflected in the Number of Children field for n^4, eliminating 0, 2, 3, and 4 as possibilities, leaving only 1.

Table 14: Fork 0 after applying InferNumChildrenForSingleton to box 2.

Fork 0					
H^0	$H^1 H^2 N^3 n^4$		$H^1 H^2 N^3 n^4$		0 1 2 3 4
H^1	H^0 $H^2 N^3 n^4$		H^0 $H^2 N^3 n^4$		0 1 2 3 4
H^2	$H^0 H^1$ $N^3 n^4$		$H^0 H^1$ $N^3 n^4$		0 1 2 3 4
N^3	$H^0 H^1 H^2$ n^4		$H^0 H^1 H^2$ n^4		0 1 2 3 4
n^4	$H^0 H^1 H^2 N^3$		$H^0 H^1 H^2 N^3$		~~0~~ 1 ~~2 3 4~~
0	1187.61	H_3Nn	$H^0 H^1 H^2 N^3 n^4$	n^4	$H^0 H^1 H^2 N^3 n^4$
1	894.43	H_3N-(ene)'	$H^0 H^1 H^2 N^3$	$H^0 H^1 H^2 N^3$	n^4
2		CH:1 PAR:0	n^4	n^4	
3	649.30	H_3-(ene)'	$H^0 H^1 H^2$	$H^0 H^1 H^2$	$N^3 n^4$
4		CH:1 PAR:1	N^3	N^3	$H^0 H^1 H^2$ n^4
5	431.19	H_2-(ene)(oh)'	$H^0 H^1$	$H^0 H^1$	$H^2 N^3 n^4$
6		CH:0 PAR:1	H^2	H^2	$H^0 H^1$ $N^3 n^4$
7	259.12	H-(oh)'	H^0	H^0	$H^1 H^2 N^3 n^4$

5.4.3.3.3 Apply Inference Rule FindRootDefinite to Box 0 and Box 2

Next the inference rule FindRootDefinite is applied to box 0 and then to box 2. This rule states that if a box has a single mono that is *possibly* the root of the box, then it is *definitely* the root of the box. In the case of both box 0 and box 2, we see that n^4 is the only possible root for either box, and so we promote it to definitely being the root of each box. We indicate this in Table 15 by the rectangle around the n^4 entry in the Root column for boxes 0 and 2.

Other inference rules will use these definite, boxed values to make further progress. By designating these values as "definite", the implementation of these other inference rules is greatly simplified.

Table 15: Fork 0 after applying FindRootDefinite to box 0 and then to box 2.

Fork 0					
H^0	$H^1 H^2 N^3 n^4$		$H^1 H^2 N^3 n^4$		0 1 2 3 4
H^1	H^0 $H^2 N^3 n^4$		H^0 $H^2 N^3 n^4$		0 1 2 3 4
H^2	$H^0 H^1$ $N^3 n^4$		$H^0 H^1$ $N^3 n^4$		0 1 2 3 4
N^3	$H^0 H^1 H^2$ n^4		$H^0 H^1 H^2$ n^4		0 1 2 3 4
n^4	$H^0 H^1 H^2 N^3$		$H^0 H^1 H^2 N^3$		1
0	1187.61	H_3Nn	$H^0 H^1 H^2 N^3 n^4$	$\boxed{n^4}$	$H^0 H^1 H^2 N^3 n^4$
1	894.43	H_3N-(ene)'	$H^0 H^1 H^2 N^3$	$H^0 H^1 H^2 N^3$	n^4
2		CH:1 PAR:0	n^4	$\boxed{n^4}$	
3	649.30	H_3-(ene)'	$H^0 H^1 H^2$	$H^0 H^1 H^2$	$N^3 n^4$
4		CH:1 PAR:1	N^3	N^3	$H^0 H^1 H^2$ n^4
5	431.19	H_2-(ene)(oh)'	$H^0 H^1$	$H^0 H^1$	$H^2 N^3 n^4$
6		CH:0 PAR:1	H^2	H^2	$H^0 H^1$ $N^3 n^4$
7	259.12	H-(oh)'	H^0	H^0	$H^1 H^2 N^3 n^4$

5.4.3.3.4 Apply Inference Rule ApplyRootDefinite to Box 0

The fact that box 0 has a definite root mono is exploited by the very next inference rule, ApplyRootDefinite. Consider box 0. It contains five monos, has a definite root (n^4), and no child scars. From this, OSCAR can infer that non-root monos in the box may each have at most 3 children. Why is this? Since n^4 is the root of this box, at least one of the other monos in the box (call it M) must attach to n^4. At that point only three monos remain in the box—even if they all attached to M, we know that M cannot have more than three children. The rule can therefore exclude 4 as a possible value for the Number of Children field for monos H^0, H^1, H^2, and N^3. See Table 16.

Table 16: Fork 0 after applying ApplyRootDefinite to box 0.

Fork 0					
H^0	H^1 H^2 N^3 n^4		H^1 H^2 N^3 n^4		0 1 2 3 4
H^1	H^0 H^2 N^3 n^4		H^0 H^2 N^3 n^4		0 1 2 3 4
H^2	H^0 H^1 N^3 n^4		H^0 H^1 N^3 n^4		0 1 2 3 4
N^3	H^0 H^1 H^2 n^4		H^0 H^1 H^2 n^4		0 1 2 3 4
n^4	H^0 H^1 H^2 N^3		H^0 H^1 H^2 N^3		1
0	1187.61	H_3Nn	H^0 H^1 H^2 N^3 n^4	$\boxed{n^4}$	H^0 H^1 H^2 N^3 n^4
1	894.43	H_3N-(ene)'	H^0 H^1 H^2 N^3	H^0 H^1 H^2 N^3	n^4
2		CH:1 PAR:0	n^4	$\boxed{n^4}$	
3	649.30	H_3-(ene)'	H^0 H^1 H^2	H^0 H^1 H^2	N^3 n^4
4		CH:1 PAR:1	N^3	N^3	H^0 H^1 H^2 n^4
5	431.19	H_2-(ene)(oh)'	H^0 H^1	H^0 H^1	H^2 N^3 n^4
6		CH:0 PAR:1	H^2	H^2	H^0 H^1 N^3 n^4
7	259.12	H-(oh)'	H^0	H^0	H^1 H^2 N^3 n^4

Consider what has happened here. This inference rule has taken multiple facts about box 0 (child scar count, contained monos, definite root mono) and updated the possible child counts for all of the non-root monos in that box. This unorthodox, seemingly unstructured data flow is the hallmark of OSCAR.

5.4.3.3.5 Apply Inference Rule ApplyLeaf to Mono H^0

We have seen several inference rules that were applied primarily to boxes. Let us now skip ahead several inference rule applications to demonstrate an inference rule that takes a mono as its input. The now-current state of fork 0 is shown in Table 17.

Notice how much progress has been made up to this point. For example, the Root and RootParent have been decided for many boxes, as evidenced by the many rectangle-enclosed monos in the table. Also, all five monos know exactly how many children they each must have: H^0 has none, H^1 has two, and so on. There is no remaining ambiguity in *child counts*, but there is in *child identity*: N^3 must have one child, but it is undecided whether that child is H^0 or H^1.

Table 17: Fork 0 after many inference rules have been applied.

Fork 0					
H^0	H^1 N^3			0	
H^1	H^0 N^3		H^0 H^2	2	
H^2	H^0 H^1			0	
N^3	n^4		H^0 H^1	1	
n^4			N^3	1	
0	1187.61	H_3Nn	H^0 H^1 H^2 N^3 n^4	n^4	
1	894.43	H_3N-(ene)'	H^0 H^1 H^2 N^3	N^3	n^4
2		CH:1 PAR:0	n^4	n^4	
3	649.30	H_3-(ene)'	H^0 H^1 H^2	H^0 H^1	N^3
4		CH:1 PAR:1	N^3	N^3	n^4
5	431.19	H_2-(ene)(oh)'	H^0 H^1	H^0 H^1	N^3
6		CH:0 PAR:1	H^2	H^2	H^0 H^1
7	259.12	H-(oh)'	H^0	H^0	H^1 N^3

From this we know that H^0 and H^2 are both leaves, that is, they have no children. Let us execute the ApplyLeaf inference rule on mono H^0 to see what further progress can be made.

This inference rule states that if mono M is a leaf, then (1) M cannot be the parent of any mono, and (2) any box containing M cannot have M as its root or root parent. We can see the effects of this inference rule in Table 18. Part (1) leads to H^0 being removed from the Parent Monos sets for H^1 and H^2; part (2) causes H^0 to be removed from the Root set for boxes 3 and 5 and from the RootParent set for box 6.

Table 18: Fork 0 after applying ApplyLeaf to mono H^0.

Fork 0			
H^0	H^1 N^3		0
H^1	~~H^0~~ N^3	H^0 H^2	2
H^2	~~H^0~~ H^1		0
N^3	n^4	H^0 H^1	1
n^4		N^3	1

0	1187.61	H_3Nn	H^0 H^1 H^2 N^3 n^4	n^4	
1	894.43	H_3N-(ene)'	H^0 H^1 H^2 N^3	N^3	n^4
2		CH:1 PAR:0	n^4	n^4	
3	649.30	H_3-(ene)'	H^0 H^1 H^2	~~H^0~~ H^1	N^3
4		CH:1 PAR:1	N^3	N^3	n^4
5	431.19	H_2-(ene)(oh)'	H^0 H^1	~~H^0~~ H^1	N^3
6		CH:0 PAR:1	H^2	H^2	~~H^0~~ H^1
7	259.12	H-(oh)'	H^0	H^0	H^1 N^3

5.4.3.3.6 Final Results for Fork 0

Inference rules are applied to fork 0 until no changes are detected after application of any rule. At this point, the fork has converged as far as OSCAR can manage. The final result for fork 0 is shown in Table 19.

Table 19: The final result for fork 0.

Fork 0					
H^0	H^1			0	
H^1	N^3		H^0 H^2	2	
H^2	H^1			0	
N^3	n^4		H^1	1	
n^4			N^3	1	
0	1187.61	H_3Nn	H^0 H^1 H^2 N^3 n^4	n^4	
1	894.43	H_3N-(ene)'	H^0 H^1 H^2 N^3	N^3	n^4
2		CH:1 PAR:0	n^4	n^4	
3	649.30	H_3-(ene)'	H^0 H^1 H^2	H^1	N^3
4		CH:1 PAR:1	N^3	N^3	n^4
5	431.19	H_2-(ene)(oh)'	H^0 H^1	H^1	N^3
6		CH:0 PAR:1	H^2	H^2	H^1
7	259.12	H-(oh)'	H^0	H^0	H^1

We see that OSCAR has successfully eliminated all ambiguity from this fork. Every mono is fully specified, having a definite parent and definite children. Every box is similarly completed, with the Root and RootParent fields all known. At this point, further applications of inference rules will make no more progress, and OSCAR will end its processing of this fork.

5.4.3.4 Calculate Scores

From the above discussion, it is clear that OSCAR needs an efficient way to detect when the inference rules are no longer making progress in simplifying a particular fork. For this, OSCAR uses the notion of scores to represent progress. A *decreasing* score implies that the fork has

been made more specific, and a score that does not change between applications of the set of inference rules means that the fork has stalled and will not benefit from further applications of the inference rules.

Each fork computes its score as a simple summation (where |X| represents the number of elements in set X):

- Add the score of each mono in the fork
- Add the score of each box in the fork

The methods mono and box use to compute their scores are very similar to each other. Recall that the **Possible** sets shrink over time and the **Definite** sets grow, and that a decreasing score implies progress is being made. OSCAR therefore adds |**Possible**| and subtracts |**Definite**| when computing the score for a mono or box. Similar reasoning is applied to the other sets contained by mono and box: the shrinking sets **NumChildrenPossible** and **Linkage** are added.

Specifically, each mono computes its score as follows:

- Add |**ParentPossible**|, |**ChildrenPossible**|, |**NumChildrenPossible**|, |**Linkage**|
- Subtract |**ParentDefinite**|, |**ChildrenDefinite**|

Each box computes its score as follows:

- Add |**RootPossible**|, |**RootParentPossible**|, |**Linkage**|
- Subtract |**RootDefinite**|, |**RootParentDefinite**|

Interestingly, a fork's score does not always correlate closely with the number of consistent glycans it produces. The score is used for only two purposes: to terminate the iterative application of the inference rules, and to prune isomorphic forks, as described in Section 5.4.3.6.

5.4.3.5 Check Consistency

After the inference rules have been applied to each fork, OSCAR examines each fork for logical inconsistencies. If any are found, the fork is marked as inconsistent and removed from the solution.

Here are some of the conditions examined to ensure that a fork is consistent, along with necessary caveats:

- Each mono must have at least one entry in its **`NumChildrenPossible`** set
    - Even if the mono is a leaf, the **`NumChildrenPossible`** set should include 0. An empty set implies that *no* integer correctly describes the number of children the mono has.
- Each mono must have at least one mono in its **`ParentPossible`** set
    - Unless the mono is possibly root of the glycan, in which case it would correctly have no parent
- Each box must have at least one mono in **`RootPossible`**
    - Because every box specifies a subtree and every subtree must have a root

Other consistency checks are performed as well, all of which verify that the constraints of a logical tree are not violated by any mono or box in the fork.

5.4.3.6 Isomorph Pruning

Selection forking can introduce forks that are *guaranteed* to generate identical sets of consistent glycans. These redundant forks are called isomorphs. Two forks are isomorphic if (1) a one-to-one mapping exists from one fork's monos to the other fork's monos, and (2) a one-to-one mapping exists from one fork's boxes to the other fork's boxes. In other words, two forks are isomorphic if one fork's monos and boxes can simply be renumbered to yield the other fork.

Consider Table 20, which represents the initial state of fork 0 while processing the pathway *m/z* 1187.6_894.4_649.2_431.1_259.0. Pay special attention to box 5, highlighted. This box required the selection of two hexose residues from a precursor that contained three (H^0, H^1, H^2). In this fork, residues H^0 and H^1 were selected for box 5. However, another fork was created where residues H^1 and H^2 were selected instead; and yet another fork where H^0 and H^2 were selected. OSCAR simply creates a fork for each possible combination.

It should be clear that the structures produced by these other forks will be the same as the ones produced by fork 0, with the exception that residues H^0/H^1 will be renumbered H^1/H^2 and H^0/H^2 in the other forks. No structural differences will be apparent, and retaining all three forks would be a waste of computational resources[7]. These three forks are isomorphs of one another, and two can be discarded. We call this process *isomorph pruning*.

---

[7] Without isomorph pruning, not only would the isomorphic forks consume resources and time, but would themselves generate more isomorphs as additional disassembly pathways were entered, and so on. OSCAR would be buried beneath an avalanche of isomorphic forks that all produce the same structures.

Table 20: The initial state of fork 0.

Fork 0					
H^0	$H^1 H^2 N^3 n^4$		$H^1 H^2 N^3 n^4$		0 1 2 3 4
H^1	H^0 $H^2 N^3 n^4$		H^0 $H^2 N^3 n^4$		0 1 2 3 4
H^2	$H^0 H^1$ $N^3 n^4$		$H^0 H^1$ $N^3 n^4$		0 1 2 3 4
N^3	$H^0 H^1 H^2$ n^4		$H^0 H^1 H^2$ n^4		0 1 2 3 4
n^4	$H^0 H^1 H^2 N^3$		$H^0 H^1 H^2 N^3$		0 1 2 3 4
0	1187.61	H_3Nn	$H^0 H^1 H^2 N^3 n^4$	$H^0 H^1 H^2 N^3 n^4$	
1	894.43	H_3N-(ene)'	$H^0 H^1 H^2 N^3$	$H^0 H^1 H^2 N^3$	n^4
2		CH:1 PAR:0	n^4	n^4	
3	649.30	H_3-(ene)'	$H^0 H^1 H^2$	$H^0 H^1 H^2$	$N^3 n^4$
4		CH:1 PAR:1	N^3	N^3	$H^0 H^1 H^2$ n^4
5	**431.19**	**H_2-(ene)(oh)'**	**$H^0 H^1$**	**$H^0 H^1$**	**$H^2 N^3 n^4$**
6		CH:0 PAR:1	H^2	H^2	$H^0 H^1$ $N^3 n^4$
7	259.12	H-(oh)'	H^0	H^0	$H^1 H^2 N^3 n^4$

A naïve implementation of isomorph pruning could compare every fork against every other fork, but that would have a run-time complexity of $O(n^2)$, a poor choice as n (the number of forks in the solution) becomes large.

Instead, OSCAR searches for isomorphic forks *after* the inference rules have been applied to all forks, which, as a side-effect, assigns a score to every fork. Because the inference rules never base their actions on the indices of the monos or boxes, isomorphic forks receive the same score. OSCAR sorts the forks by score, puts forks with identical scores into buckets, and then searches for isomorphic pairs only within each bucket. This implementation of isomorph pruning is critical in achieving OSCAR's fast execution times.

5.4.3.7 The Summarize Command

When the **`Summarize`** command is issued, OSCAR generates and displays a set of glycans consistent with all selected disassembly pathways. OSCAR then displays some solution statistics. An overview of this process is shown in Figure 26.

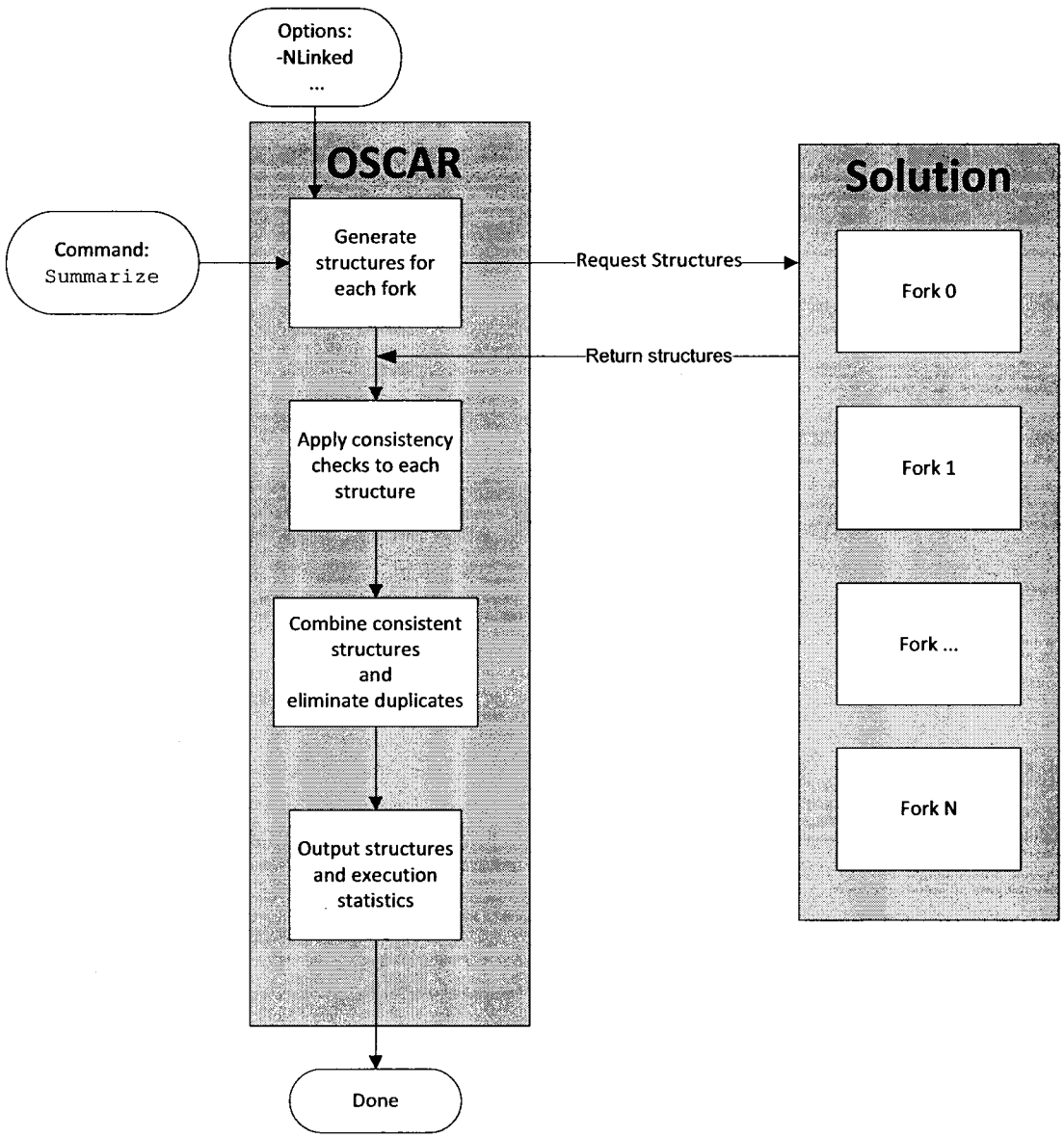


Figure 26: A simplified view of the Summarize command.

5.4.3.7.1 Generate Consistent Glycans

This phase generates the output that the analyst is most interested in: the complete set of consistent glycans as restricted by the selected MS^n pathways.

To do this, OSCAR iterates over all remaining forks in the solution. For each fork, the **`ParentPossible`** set for each Mono is examined. A candidate glycan is assembled with every mono attached to one of its possible parents. Then the glycan is subjected to further consistency checks (more below) and, if found to be valid, displayed. This process iterates until every possible mono parent-child combination, and therefore every possible glycan, is examined.

Each candidate glycan is subjected to these consistency checks, and more, before it is considered to be consistent:

- Every mono in the glycan should be reachable from the root of the proposed glycan
- The number of children any mono has must be consistent with that Mono's **`NumChildrenPossible`**
- If the mono has any definite (instead of merely possible) children, then that parent-child relationship must be present in the proposed glycan
- For every box in the fork:
    - The monos in that box must form a connected subtree embedded in the proposed glycan
    - The embedded subtree must have the same number of child scars (children of the glycan that fall outside the box) as the box
    - The root Mono's **`Linkage`** must agree with the Box's **`Linkage`**

These checks can be thought of as a second line of constraints, above and beyond those that manipulate the fork, mono and box data structures.

For the input example shown in Listing 4 on page 60, the single topology output matches the expected glycan of Figure 24. GlySpy's text output of glycan structures follows the time-honored computer science approach of indenting to indicate a child relationship. Additionally, the parenthesized numbers following each node represent the possible linkage of that node to its parent. Also given are two linear code representations of the structure, one with interresidue linkage details and one without. See Listing 6.

```
Generated 1 unique glycan (from 1 consistent, 1 total):
======== Begin 1 Unique Glycan ========
- - - - Begin unique glycan - - - -
Linear code (branching): H(H)HNn
Linear code (w/linkage): H(H)Hx346Nx346n
n4 ()
  N3 (346)
    H1 (346)
      H2 (2346)
      H0 (2346)
Glycan is supported by 1 specific Fork/Glycan pair:
0/0
- - - - End unique glycan - - - -
======== End 1 Unique Glycan ========
```

Listing 6: GlySpy's output for the topology of the five residue *N*-linked core.

5.4.3.7.2 Display Execution Statistics

Following the set of consistent glycans, OSCAR displays some execution statistics. For the example being discussed, some of the statistics displayed are shown in Listing 7.

```
Total forks: 7    Live: 1    Dead: 6  (Inconsistent: 1  Isomorph: 5)
RunInferencesOnce called 27 times
```

Listing 7: OSCAR execution statistics.

This shows that over the course of the program's execution, a total of seven forks were created, but six of them were removed from the solution, leaving a lone consistent fork. Of the

six removed forks, one was inconsistent and five were isomorphs. We also see that the full set of inference rules was applied a total of 27 times. The –time command-line switch causes this additional output:

```
GlySpyCLI: Total elapsed time: 0.01 secs
```

5.5. Results for a Fourteen-Residue Glycan ($H_6N_4S_3n$, *m/z* 3618.8)

Now we apply OSCAR to a larger glycan, specifically, a 14-residue glycan isolated from fetal calf fetuin, an important blood glycoprotein. As shown in Figure 27, this glycan's *N*-linked core (residues $H^0/H^1/H^2/N^6/n^{13}$) is decorated with three separate SHN antennae ($S^{10}/H^3/N^7$, $S^{11}/H^4/N^8$, and $S^{12}/H^5/N^9$). In this section we demonstrate how an analyst might select observed MS^n pathways to allow OSCAR to compute this glycan's branching topology, SHN(SHN)H'(SHNH')H'N'n'. To our knowledge, *de novo* topology analysis has not previously been reported for any glycan of this size.

A simplified view of the glycan is shown in Figure 28.

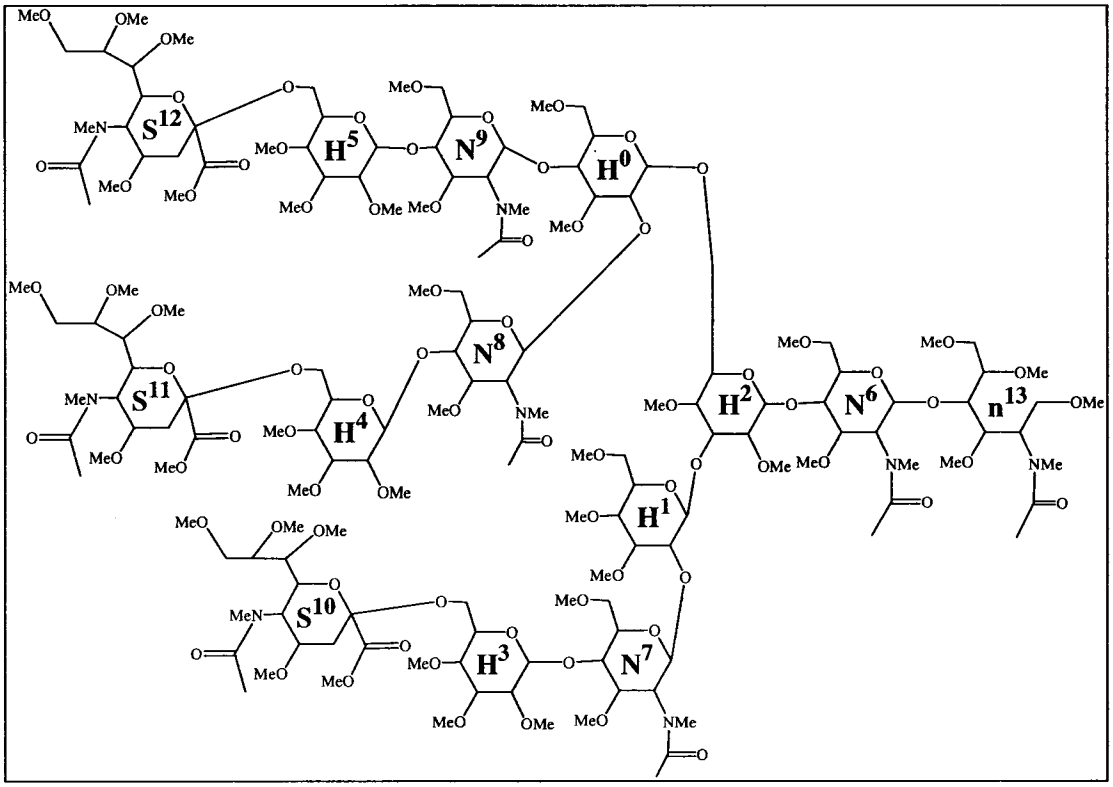


Figure 27: A tri-sialylated glycan (*m/z* 3618.8) as isolated from fetuin.

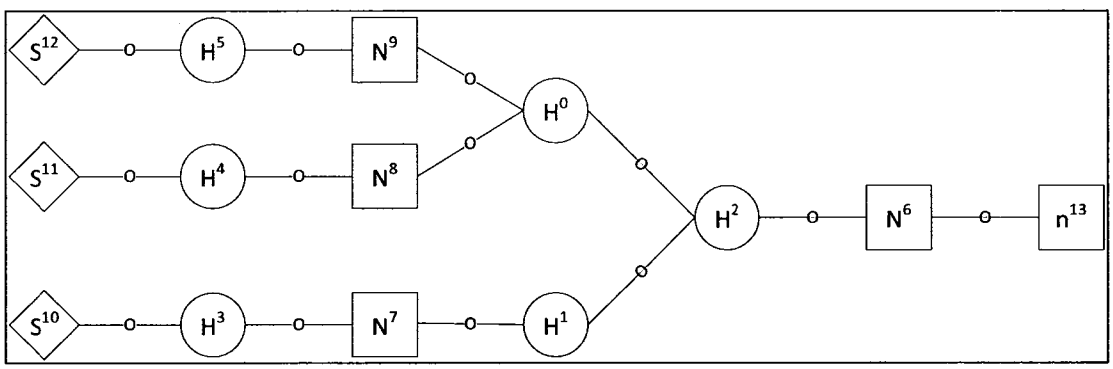


Figure 28: Simplified diagram of glycan *m/z* 3618.8.

For a variety of reasons, this glycan is difficult to analyze by MS^n alone. It has three identical SHN antennae (which conspire to confuse the analysis), has a large number of facile cleavages (which tend to absorb the collision energy and limit the variety of fragments generated), and is a large structure (even doubly-charged, its *m/z* of 1820.9^{2+} is near the 2000 Da limit of the LTQ mass spectrometer).

We first present the idealized fragmentation pattern one might expect from this glycan, and how the analyst might select from these pathways for use by OSCAR. Then we revisit this glycan using actual data, to show the trade-offs an analyst must make attempting a *de novo* analysis of a large structure.

What types of fragmentation should we expect from this structure? The glycosidic bonds originating from HexNAc residues (N) are generally the weakest bonds in the glycan and tend to rupture most easily. The bonds from sialic acid residues (S) are also quite weak. These heuristics can be affected by the exact conformation of the glycan, but serve as reasonable guides. As such, we would expect cleavages at the reducing end of residues N^6, N^7, N^8, N^9, with additional cleavages at the reducing ends of residues S^{10}, S^{11}, and S^{12}. (That is, from Figure 28, we expect the bonds to the *right* of N and S residues to break relatively easily.) This in fact is largely what we observe in Spectrum A-1 on page 208 of the appendix. The peaks of this spectrum are detailed in Table 21.

The charge state of an ion can be reliably inferred from the spacing of peaks in the isotopic window. Spectrum A-2 details the isotopic envelope for *m/z* 847.4. This series of peaks is caused by the random inclusion of ^{13}C isotopes, instead of the more common ^{12}C, which causes a shift of one mass unit. Because the peaks are separated by intervals of 1 *m/z*, the charge state is known to be +1. (We know m changes by one, and so z must also be one to register an *m/z* difference of one.) Spectrum A-3 shows several peaks separated by approximately 0.5 mass units (1258.1, 1258.5 and 1259.0; 1262.0, 1262.5 and 1263.1) revealing a charge state of +2. (Again, m changes by one, but z = 2 gives *m/z* differences of 1/2.) This spectrum also shows that monoisotopic peaks (1258.1 and 1262.0) may not be the most intense in the envelope. Careful interpretation is required to assign charge states and ascertain monoisotopic masses.

Table 21: Ions observed on the spectrum for fetuin *m/z* 1820.9^{2+} ($H_6N_4S_3n$).

Observed *m/z*	Charge State	Singly-Charged *m/z*	Most Likely Composition	Theoretical *m/z*	Description
847.4	+1	847.4	HNS-(ene)'	847.41	Any SHN antenna
1221.1	+2	2419.21	H_5N_3Sn-$(oh)_2$	2419.21	Loss of SHN and S
1258.1	+2	2493.21	H_6N_4n-$(oh)_3$	2493.25	Loss of all three S
1262.0	+2	2501.01	$H_5N_3S_2$-(ene)(oh)'	2501.22	Loss of SHN and n
1299.1	+2	2575.21	H_6N_4S-(ene)$(oh)_2$'	2575.25	Loss of two S and n
1408.6	+2	2794.21	$H_5N_3S_2n$-(oh)	2794.40	Loss of SHN antenna
1445.6	+2	2868.21	H_6N_4Sn-$(oh)_2$	2868.44	Loss of two S
1486.6	+2	2950.21	$H_6N_4S_2$-(ene)(oh)'	2950.44	Loss of S and n
1633.2	+2	3243.41	$H_6N_4S_2n$-(oh)	3243.63	Loss of S
1674.3	+2	3325.61	$H_6N_4S_3$-(ene)'	3325.63	Loss of n

All of the fragments in this table can be explained by reducing-end cleavages of one or more N or S residues. The lone exception from Spectrum A-1, minor ion *m/z* 1805.3, appears to be caused by electronic noise. Spectrum A-4 shows detail near *m/z* 1805.3; the nature of this signal is consistent with electronic noise periodically experienced by the Glycomics Center's LTQ mass spectrometer. Spectra in this work were often selected for their avoidance of the high-*m/z* area where this noise is most prevalent.

These results suggests a strategy for approaching the glycan with OSCAR: cleave off an SHN antenna, sequence the antenna, and then repeat for the second and third antennae. The expected fragments for this strategy are shown in Figure 29 through Figure 32.

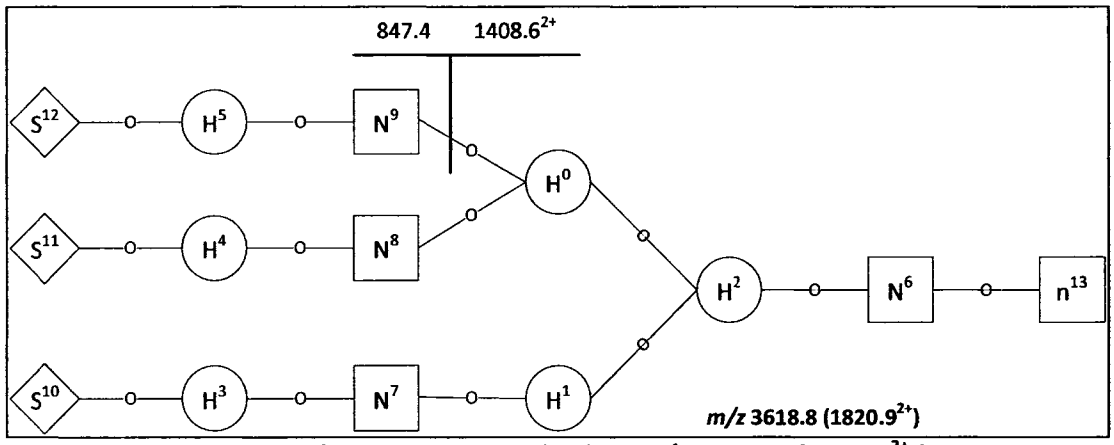


Figure 29: Cleavage of one SHN antenna leads to *m/z* 847.4 and 1408.6^{2+} fragments.

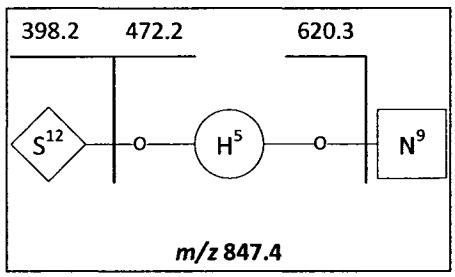


Figure 30: A few expected fragments from one SHN antenna.

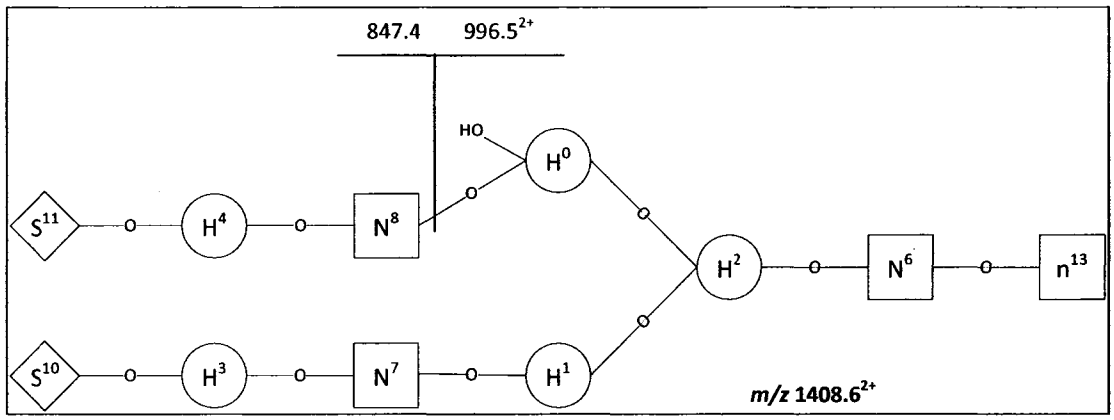


Figure 31: Cleavage of a second SHN antenna.

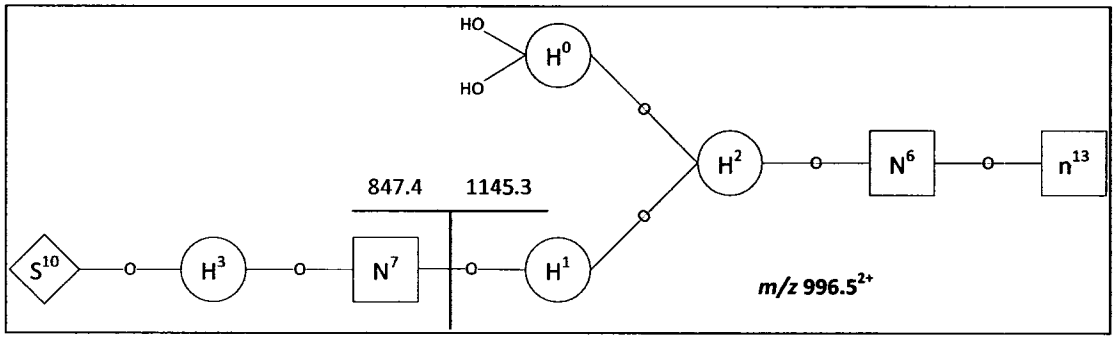


Figure 32: Cleavage of the final SHN antenna.

At this point, OSCAR will have determined that three separate SHN antenna exist, but will not yet know where they are located on the *N*-linked core. The analyst might then look for the loss of the reducing-end n residue (Figure 33) followed by an analysis of the remaining H_3N core residues (Figure 34). Such a theoretical strategy is in fact successful and leads to the GlySpy input shown in Listing 8. This input produces a single correct branching topology, SHN(SHN)H'(SHNH')H'N'n'. Many other successful strategies also exist, though OSCAR may process them more or less efficiently than this one.

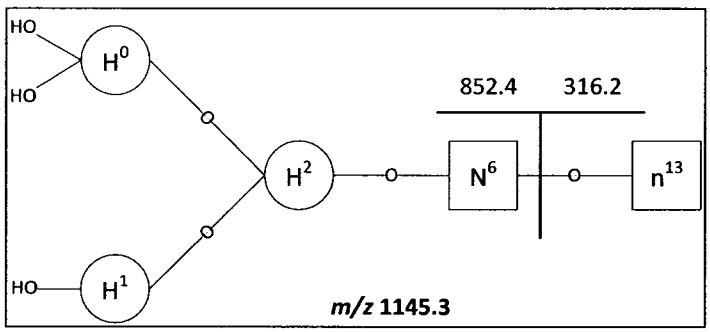


Figure 33: Cleavage of the reducing-end n residue.

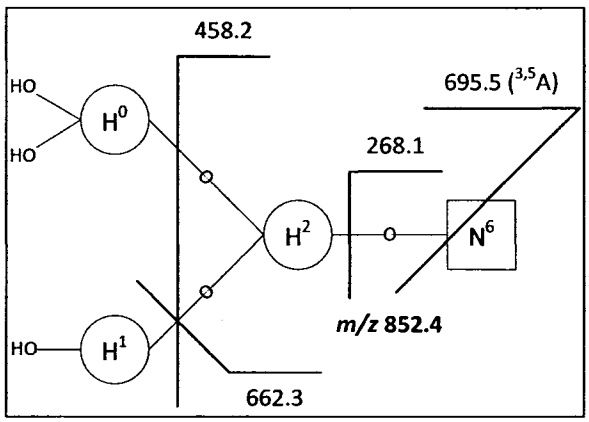


Figure 34: Some expected fragments of the H_3N *N*-linked core.

```
-NLinkedBranching

; Treat each lost branch as complementary to the remaining residues
-DisjointComplements

; Lose three SHN- branches in sequence
AddPathway NoCrossRing 1820.9x2_1408.6x2_996.5x2_1145.3

; Sequence first SHN- branch (complementary to 1820.9x2_1408.6x2)
AddPathway NoCrossRing 1820.9x2_847.4_620.1
AddPathway NoCrossRing 1820.9x2_847.4_398.1

; Second SHN- branch (complementary to 1820.9x2_1408.6x2_996.5x2)
AddPathway NoCrossRing 1820.9x2_1408.6x2_847.4_620.1
AddPathway NoCrossRing 1820.9x2_1408.6x2_847.4_398.1

; Third SHN- branch (complementary to 1820.9x2_1408.6x2_996.5x2_1145.3)
AddPathway NoCrossRing 1820.9x2_1408.6x2_996.5x2_847.4_620.1
AddPathway NoCrossRing 1820.9x2_1408.6x2_996.5x2_847.4_398.1

; Lose reducing-end n
AddPathway NoCrossRing 1820.9x2_1408.6x2_996.5x2_1145.3_852.4

; 852 represents the H3N N-linked core (minus the reducing end n).
AddPathway NoCrossRing 1820.9x2_1408.6x2_996.5x2_1145.3_852.4_458.4
AddPathway NoCrossRing 1820.9x2_1408.6x2_996.5x2_1145.3_852.4_662.2

Summarize
```

Listing 8: A successful disassembly strategy based on expected fragments.

Instrument sensitivity limitations make some theoretical spectra unobtainable in practice, and so alternate ions must be selected by the analyst. Listing 9 shows a sequencing strategy supported by the experimental data (Spectrum A-1, plus Spectrum A-5 through Spectrum A-9).

```
-NLinkedBranching

; Treat each lost branch as complementary to the remaining residues
-DisjointComplements

; Lose three SHN- branches in sequence
AddPathway NoCrossRing 1820.9x2_1408.6x2_996.5x2_1145.3

; Sequence first SHN- branch (complementary to 1820.9x2_1408.6x2)
AddPathway NoCrossRing 1820.9x2_847.4_620.1
AddPathway NoCrossRing 1820.9x2_847.4_398.1

; Second SHN- branch (complementary to 1820.9x2_1408.6x2_996.5x2)
AddPathway NoCrossRing 1820.9x2_1408.6x2_847.4_620.1
AddPathway NoCrossRing 1820.9x2_1408.6x2_847.4_398.1

; Third SHN- branch (complementary to 1820.9x2_1408.6x2_996.5x2_1145.3)
AddPathway NoCrossRing 1820.9x2_1408.6x2_996.5x2_847.4
AddPathway NoCrossRing 1820.9x2_1408.6x2_996.5x2_602.1
AddPathway NoCrossRing 1820.9x2_1408.6x2_996.5x2_472.1
AddPathway NoCrossRing 1820.9x2_1408.6x2_996.5x2_398.1

; 852 represents the H3N N-linked core (minus the reducing end n).
AddPathway NoCrossRing
 » 1820.9x2_1633.3x2_1445.8x2_1258.0x2_1033.5x2_887.0x2_1301.5_852.3

AddPathway
 » 1820.9x2_1633.3x2_1445.8x2_1258.0x2_1033.5x2_887.0x2_1301.5_852.3_695.5

AddPathway NoCrossRing
 » 1820.9x2_1633.3x2_1445.8x2_1258.0x2_1033.5x2_887.0x2_1301.5_852.3_458.4

AddPathway NoCrossRing
 » 1820.9x2_1633.3x2_1445.8x2_1258.0x2_1033.5x2_887.0x2_1301.5_852.3_662.2

Summarize
```

Listing 9: A successful disassembly strategy based on experimental data. The » symbols represent line breaks inserted for formatting purposes.

There are two main differences moving from Listing 8 to Listing 9:

1) The third SHN branch cannot be sequenced through these two pathways:

```
AddPathway NoCrossRing 1820.9x2_1408.6x2_996.5x2_847.4_620.1
AddPathway NoCrossRing 1820.9x2_1408.6x2_996.5x2_847.4_398.1
```

because the ion *m/z* 847 is not intense enough to fragment. The normalization level of the precursor (Spectrum A-6) is already quite low, at 8.63E-1. Instead of isolating and fragmenting ion *m/z* 847, the analyst notices that the desired ions are available in the precursor spectrum and uses them directly. Because of the lack of context, however, more ions are required:

```
AddPathway NoCrossRing 1820.9x2_1408.6x2_996.5x2_847.4
AddPathway NoCrossRing 1820.9x2_1408.6x2_996.5x2_602.1
AddPathway NoCrossRing 1820.9x2_1408.6x2_996.5x2_472.1
AddPathway NoCrossRing 1820.9x2_1408.6x2_996.5x2_398.1
```

2) The theoretical pathway to the *m/z* 852 *N*-linked core residues (Figure 34) was

```
AddPathway NoCrossRing 1820.9x2_1408.6x2_996.5x2_1145.3_852.4_458.4
AddPathway NoCrossRing 1820.9x2_1408.6x2_996.5x2_1145.3_852.4_662.2
```

However, again, the low intensity of the 996.5^{2+} spectrum forces the analyst to use a different strategy. In this case, a higher-intensity pathway to the core residues is used:

```
AddPathway NoCrossRing
 » 1820.9x2_1633.3x2_1445.8x2_1258.0x2_1033.5x2_887.0x2_1301.5_852.3

AddPathway
 » 1820.9x2_1633.3x2_1445.8x2_1258.0x2_1033.5x2_887.0x2_1301.5_852.3_695.5

AddPathway NoCrossRing
 » 1820.9x2_1633.3x2_1445.8x2_1258.0x2_1033.5x2_887.0x2_1301.5_852.3_458.4

AddPathway NoCrossRing
 » 1820.9x2_1633.3x2_1445.8x2_1258.0x2_1033.5x2_887.0x2_1301.5_852.3_662.2
```

Here we see a common analytical trick. The ion *m/z* 852.3 is established by the first command, and then ion *m/z* 695.5 is given next and interpreted as a cross-ring fragment. (Note that the second AddPathway command does not specify the NoCrossRing option.) This ion, a loss of 157 mass units, represents a ^{3,5}A cleavage of a reducing-end HexNAc (specifically, residue

N^6), and specifies that the *m/z* 852 fragment under consideration must have a reducing-end N. The analyst then selects the familiar *m/z* 458.4 and 662.2 ions to complete the assignment.

Listing 9 also produces a single correct structure, although more slowly than the idealized Listing 8. See Table 22. However, we have shown that GlySpy is capable of *de novo* analysis of relatively large glycans without resorting to biosynthetic rules, database lookups, or spectrum fingerprint matching. The structure generated results strictly from the *N*-linked core motif coupled with the constraints derived from the selected pathways.

Input	Execution Time (seconds)
Listing 8	0.27
Listing 9	4.39

Table 22: Execution times for the input shown in Listing 8 and Listing 9.

5.6. Limitations/Future Work

OSCAR has proven to be a capable tool for expert-level analysts. However, several enhancements would increase its utility.

First, and most obviously, a friendly graphical user interface (GUI) would enable non-expert analysts to derive more benefit from the tool.

Next, facile N and S cleavages could be used as further constraints. For example, if N and/or S residues are present in a precursor, all the major products should be considered as resulting from parent cleavages of those residues.

Last, auto assignment of charge states and monoisotopic peaks would be of great value. Removing this tedious step would greatly improve the analyst's productivity. However, any

automated spectral analysis would need to deal with a variety of practical considerations: Electronic noise, low normalization levels, poor instrument calibration, and so on.

More speculatively, there may be other problem domains amenable to OSCAR's computational approach. One possible example is the derivation of phylogenetic trees from a series of individual genomes. Assuming that all represented species can be placed within an evolutionary tree, OSCAR's technology might be adapted to discover the exact shape of that tree. Current approaches in comparative genomics often group species into subtrees whose exact internal topology is incompletely known; these groups might map well to OSCAR's box data structure and subsequent processing.

Beyond being an expert-level tool, OSCAR has become a building block for the other main GlySpy algorithms: IsoDetect, IsoSolve, and Intelligent Data Acquisition. These are covered in following chapters.

5.7. Discussion

A few topics beyond OSCAR's implementation and analytical results are worth brief mention.

5.7.1. Comparisons with Algorithm Archetypes

OSCAR shares some characteristics of various well-known algorithm archetypes. At its core, OSCAR maintains a set of directional graphs that represent possible connectivity between monosaccharide residues. It applies constraint-based rules derived from fragmentation pathways to remove links between nodes in these graphs. The rules are based on the logical properties of trees, and are applied iteratively until no further links can be severed. On request, the algorithm extracts the set of trees that are embedded in these graphs; these trees are the glycans that are consistent with all selected fragmentation pathways.

Given this, we can now very briefly compare OSCAR to expert systems, constraint-based systems, and blackboard systems. This discussion is not meant to be exhaustive, but rather to point out OSCAR's similarities and differences with these architectures. Much of this section is based upon (Russell and Norvig 74).

5.7.1.1 Expert Systems

Expert systems encode a large amount of very domain-specific knowledge in the attempt to match the performance of a human expert.

DENDRAL (Buchanan 12; Feigenbaum 27; Lederberg 56; Lindsay 57) is generally regarded as one of the first expert systems. Coincidentally, it was designed to infer molecular structure from mass spectra! The input was the elementary formula for the compound and a spectrum of fragments generated by electron bombardment. Rules were determined through extensive consultation with analytical chemists. One example rule (Russell and Norvig 74) is given to identify a ketone group (C=O, mass: 28 Da) in a molecule whose mass is M Da:

```
if there are two peaks at x1 and x2 such that:
      (a) x1 + x2 = M + 28
      (b) x1 - 28 is an intense peak
      (c) x2 - 28 is an intense peak
      (d) At least one of x1 and x2 is an intense peak
then there is a ketone subgroup present.
```

The rule encodes the case where peaks x1 and x2 represent different fragments of the precursor molecule, but where x1 and x2 both contain the ketone group. The x1-28 and x2-28 peaks represent those fragments both losing the ketone group.

It could be argued that the tree-based constraint rules at OSCAR's core comprise a type of expert system. The rules are generally short and self-contained, similar to the one shown above. As a simple example, consider the ApplyLeaf inference rule from Section 5.4.3.3.5 on page 73. Here, OSCAR has determined that a mono has no possible children, that is, it

represents a terminal residue. Clearly the mono is a leaf and cannot be the parent of any mono in the graph. OSCAR sweeps over all monos and removes the leaf from the possible parent list of each (and performs several other leaf-based restrictions as well).

There is a clear difference in the domains in which DENDRAL and OSCAR can be considered experts. DENDRAL encodes chemical and spectral rules, whereas OSCAR encodes tree-based rules[8].

5.7.1.2 Constraint-Based Systems

The problem OSCAR intends to solve can be viewed as a naïve constraint satisfaction problem—see, for example, Chapter 5 of (Russell and Norvig 74). Given multiple fragmentation pathways as inputs, a program could generate all possible ion composition interpretations, and then screen all possible glycans against them to rule out inconsistent structures. (Such a tool, STAT, is discussed in Section 4.8 on page 41.) Obviously, such an implementation is doomed to failure given larger glycans. Even the capabilities of advanced generic constraint-based systems might be overwhelmed by the explosive growth in the size of the problem space.

However, OSCAR performs a highly optimized version of this computation. It does indeed start with all possible composition interpretations, but performs significant pruning of these possibilities. It is also capable of generating all possible glycan structures for a given starting composition, but again manages its data structures very carefully to improve its performance. As this chapter has shown in some detail, OSCAR goes to great lengths to avoid the combinatorial problems associated with glycan analysis.

---

[8] Recall that the inference rules seen in Section 5.4 were based almost entirely on the properties of trees, not on the specific chemical or spectral properties of glycans.

5.7.1.3 Blackboard Systems

In a blackboard architecture, the current state of a partially-solved problem is kept on a shared data structure known as a *blackboard.* Multiple *experts* are implemented as logically separate modules, and each given a chance to contribute what they can to the overall solution. Experts can read, add, change, or even remove information from the blackboard, which leads to a rich, but unpredictable, interaction between the experts. Much work has been done to guarantee that blackboard systems will converge on a solution, or at least declare that one is not forthcoming, instead of becoming stuck in an endless reasoning cycle. The choice of which expert to execute next is a crucial design decision that is often difficult for designers to implement without significant experience with the system.

OSCAR can be viewed as a very specialized blackboard system. In this view, the solution data structure and its contained forks are the blackboard and each inference rule is a miniature expert. The crucial difference, though, is that OSCAR's "experts" are constrained from making changes that could lead to cyclic reasoning, guaranteeing that the algorithm will terminate. (See Section 5.7.2.)

OSCAR's selection of which expert to execute next is also greatly simplified: they are all tried in order, iteratively, until no further changes to the data structures are made.

5.7.2. Algorithm Termination

Consider the question of algorithm termination: How do we know that OSCAR will always terminate regardless of the selected input pathways? A brief explanation is warranted here.

Recall from Section 5.3 on page 53 that OSCAR maintains **`Possible`** and **`Definite`** sets of parent and child monos. A key property of every inference rule is that **`Possible`** sets are only allowed to *shrink* and **`Definite`** sets are only allowed to *grow*. We see that progress is

monotonic: it is never possible for two rules to get caught in an endless battle of adding and then removing the same elements. Because these rules are applied iteratively until no changes in the sets are observed, OSCAR is guaranteed to terminate.

5.8. Summary

We have seen how OSCAR processes analyst-selected MS^n pathways to propose glycan topologies. In the following chapters, we will learn how GlySpy's higher-level tools—IsoDetect, IsoSolve, and Intelligent Data Acquisition—build upon OSCAR to perform their functions.

CHAPTER 6:

ISODETECT

6.1. Overview

Analysts can easily be overwhelmed by the volume of data contained in even a handful of spectra. IsoDetect has been developed to automatically determine which disassembly pathways are consistent with (and which are inconsistent with) a set of expected glycan structures. The set of consistent pathways neatly summarizes the evidence in support of the expected structures, but the inconsistent pathways are typically more prized, as they may indicate the presence of previously-unreported structural isomers. This serves to focus the analyst's attention on the pathways most likely to lead to the assignment of these alternative structures. This "isomer detection" capability gives IsoDetect its name.

More specifically, IsoDetect accepts a set of raw spectral files (the ".raw" files produced by the Thermo Fisher LTQ mass spectrometer) along with a list of structures expected to be found at a given mass, and outputs a summary of the disassembly pathways found and the structures, if any, with which each pathway is consistent.

6.2. Commands

To input the raw spectral files to examine, the analyst uses the AddSpectrumFile command discussed in Section 5.2.3 on page 51. Other relevant commands include the AddProposedGlycan and IsoDetect commands.

6.2.1. The AddProposedGlycan Command

The analyst uses the linear code notation described in Section 3.7 to describe each expected structure. As an example, consider the GM1a and GM1b glycoconjugates from Figure 11 on page 16. To add these two structures to the set of expected glycans, the analyst would issue to commands:

```
AddProposedGlycan HN(S)HH-(oh)        ; GM1a
AddProposedGlycan SHNHH-(oh)          ; GM1b
```

As indicated by the comments, the first command represents the topology of GM1a and the second, GM1b.

6.2.2. The IsoDetect Command

After using the **AddSpectrumFile** and **AddProposedGlycan** commands to provide input, the **IsoDetect** command may be issued. It has the form:

```
IsoDetect [NoCrossRing] MZ-target rel-intensity XML-output
```

The **NoCrossRing** option restricts processing to those input pathways that can be described by glycosidic fragmentations only. The **MZ-target** parameter gives the observed *m/z* for the target glycans, **rel-intensity** specifies a relative intensity cut-off below which pathways are ignored, and **XML-output** specifies the path to an optional XML file that contains the command's output in a machine-readable format. The XML format is not covered by this

document and all examples will use "nul" as the **XLM-output** parameter, meaning that no file should be written.

Listing 10 shows sample IsoDetect input where only GM1a is given as an expected structure; Listing 11 provides both GM1a and GM1b as expected structures. In both listings, the spectrum files used are those collected automatically via Intelligent Data Acquisition. (The process of using IDA to collect these particular spectra is discussed in Section 9.2.1.1 on page 160.) Also in both cases, a relative intensity cut-off of 2% is used.

These listings will be used as the basis for discussion of the IsoDetect algorithm (Section 6.3) and its results (Section 6.4).

```
-ReducingEndResidue unreduced
-UnmethylatedReducingEnd

; Add raw spectral data files
AddSpectrumFile GM1ab_1877_1273.raw
AddSpectrumFile GM1ab_1877_1273_898.raw
AddSpectrumFile GM1ab_1877_1273_898_486.raw
AddSpectrumFile GM1ab_1877_1273_847.raw
AddSpectrumFile GM1ab_1877_1273_898_435.raw
AddSpectrumFile GM1ab_1877_1273_847_472.raw
AddSpectrumFile GM1ab_1877_1273_898_449.raw
AddSpectrumFile GM1ab_1877_1273_898_472.raw

; Add topology for just one of the expected structure (GM1a)
AddProposedGlycan HN(S)HH-(oh)       ; GM1a

; Run IsoDetect, but do not generate the XML file (output file is nul)
IsoDetect NoCrossRing 1273.65 2 nul
```

Listing 10: IsoDetect input where only GM1a is given as an expected structure.

```
-ReducingEndResidue unreduced
-UnmethylatedReducingEnd

; Add raw spectral data files
AddSpectrumFile GM1ab_1877_1273.raw
AddSpectrumFile GM1ab_1877_1273_898.raw
AddSpectrumFile GM1ab_1877_1273_898_486.raw
AddSpectrumFile GM1ab_1877_1273_847.raw
AddSpectrumFile GM1ab_1877_1273_898_435.raw
AddSpectrumFile GM1ab_1877_1273_847_472.raw
AddSpectrumFile GM1ab_1877_1273_898_449.raw
AddSpectrumFile GM1ab_1877_1273_898_472.raw

; Add topologies for both expected structure (GM1a and GM1b)
AddProposedGlycan HN(S)HH-(oh)      ; GM1a
AddProposedGlycan SHNHH-(oh)        ; GM1b

; Run IsoDetect, but do not generate the XML file (output file is nul)
IsoDetect NoCrossRing 1273.65 2 nul
```

Listing 11: IsoDetect input where both GM1a and GM1b are given as expected structures. The AddSpectrumFile commands are identical to those in Listing 10.

6.3. Algorithm

IsoDetect accepts a list of expected structures, plus a set of spectral data files from which it extracts all pertinent disassembly pathways. IsoDetect's goal, then, is to determine the consistency of every pathway/structure pair. Intuitively, a pathway and structure are *consistent* if some sequential disassembly of the structure yields all of the ions in the pathway.

IsoDetect is implemented as a layer above OSCAR. To determine the consistency of one pathway/structure pair, OSCAR first creates a solution containing one fork (Section 5.3.1 on page 54) but instead of initializing the fork to represent *all possible structures*, OSCAR initializes it to represent *only the expected structure*. For the GM1a and GM1b structures specified in Listing 11, IsoDetect would initialize two forks as shown in Table 23 (for GM1a) and Table 24 (for GM1b). Notice that every parent/child relationship has been definitively assigned, as indicated by the boxed text, and that the root of the glycan is known to be H^0.

Table 23: The IsoDetect fork as initialized to match GM1a's branching topology.

Fork for GM1a					
H^0			H^2		1
H^1	N^3				0
H^2	H^0		N^3 S^4		2
N^3	H^2		H^1		1
S^4	H^2				0
0	1273.62	H_3NS-(oh)'	H^0 H^1 H^2 N^3 S^4	H^0	

Table 24: The IsoDetect fork for GM1b's branching topology.

Fork for GM1b					
H^0			H^2		1
H^1	N^3		S^4		1
H^2	H^0		N^3		1
N^3	H^2		H^1		1
S^4	H^1				0
0	1273.62	H_3NS-(oh)'	H^0 H^1 H^2 N^3 S^4	H^0	

The pathway then is analyzed for compatibility with this expected structure, using OSCAR's normal method of recognizing and applying the structural constraints imposed by each pathway. IsoDetect adds the pathway to the solution and then applies constraints exactly as if **`AddPathway`** and **`Summarize`** commands had been given. If the application of OSCAR's

inference rules leaves at least one fork alive in the solution[9], the pathway/structure pair is considered to be consistent. However, if no live forks remain in the solution, the pathway/structure pair is inconsistent.

This process is repeated until each pathway has been tested for consistency against each expected structure. See Figure 35, which illustrates the processing done for the GM1a/GM1b example of Listing 11. Notice that separate solutions are created, in turn, to represent GM1a and GM1b, and each pathway is tested against these solutions.

[9] Notice that adding the pathway to the solution can cause forks to be created, for example, through selection forking (Section 5.4.3.2.2 on page 70). This ensures that all possible interpretations of the input pathway are considered.

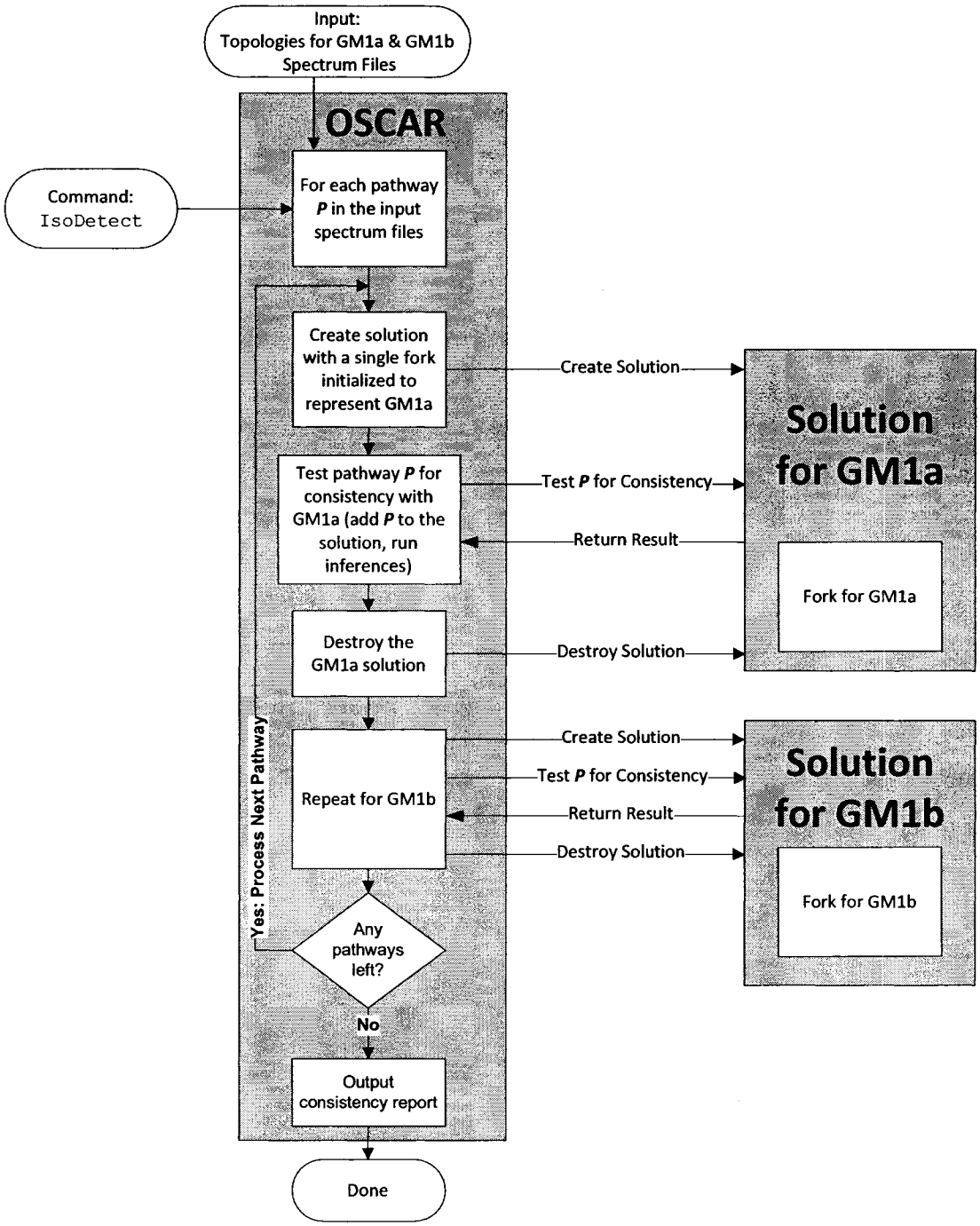


Figure 35: IsoDetect processing of the GM1a/GM1b example from Listing 11.

6.4. Results

6.4.1. Results for GM1a/GM1b

We now look at IsoDetect's results given the input from Listing 10, where only GM1a is given as an expected structure, and Listing 11, where both GM1a and GM1b are given.

6.4.1.1 GM1a Only (Listing 10)

IsoDetect extracts a total of 32 fragmentation pathways for analysis. Of these, 15 are deemed to be consistent with GM1a and 17 are inconsistent. One example of each type of pathway is found directly on Spectrum A-10 (see page 213), the MS^2 spectrum for the GM1a/GM1b isomeric mixture.

Pathway *m/z*	Compositions	Consistent with GM1a?
1273.40_435.03	1273.62 H_3NS-(oh)' 435.18 H_2-$(oh)_3$'	Yes
1273.40_449.14	1273.62 H_3NS-(oh)' 449.20 H_2-$(oh)_2$'	No

Table 25: Two selected pathways from the GM1a/GM1b mixture. The first is consistent with GM1a, the second is not.

Looking again at Figure 11 on page 16, we can see that the pathway terminated by ion *m/z* 435 is easily explained by GM1a. Because its composition has two hexoses and three scars, this pathway maps neatly to residues H^0 and H^2. These hexoses would require two cleavages to be removed from the structure (cleaving above N^3 and S^4), with the third scar coming from the open hydroxyl at the glycan's reducing end.

The second pathway, with composition 449.20 H_2-$(oh)_2$', does not fit GM1a. There is no way to cleave a pair of connected hexoses from GM1a to yield a fragment with only two scars. This

pathway, along with many others in the IsoDetect report, is strong evidence that a structural isomer is present.

6.4.1.2 GM1a and GM1b (Listing 11)

Now assume that the analyst has successfully identified both GM1a and GM1b. Are there still pathways that point at the possibility of yet another structural isomer? This is answered by IsoDetect's output for Listing 11, as summarized in Table 26. Here all 32 processed pathways are given, along with the composition assigned to each ion in each pathway. For brevity, the initial ion *m/z* 1273.62 H_3NS-(oh)' is omitted from each pathway.

Reading down the second column, we see that 1273.62_435.18 is consistent with GM1a (but not GM1b), as are 1273.62_486.23, 1273.62_810.37, and so on through to 1273.62_898.43_435.18_213.08, for a total of 13 consistent pathways. The next column shows that only two observed pathways–1273.62_898.43 and 1273.62_898.43_676.32–are compatible with both GM1a and GM1b. The last column shows 17 different pathways consistent with GM1b (but not GM1a).

Significantly, the combination of GM1a and GM1b explains every observed pathway (13 + 2 + 17 = 32 pathways). Other structural isomers may still be present, perhaps hidden due to low abundance or a clever structure that mimics parts of GM1a and GM1b, but Table 26 is strong evidence that GM1a and GM1b are likely the only isomers present down to the 2% relative intensity threshold selected by the analyst.

#	Pathways Consistent with GM1a Only	Pathways Consistent with Both GM1a and GM1b	Pathways Consistent with GM1b Only
1	435.18 H_2-(oh)$_3$'	898.43 H_3N-(oh)$_2$'	449.20 H_2-(oh)$_2$'
2	486.23 HN-(ene)'	898.43 H_3N-(oh)$_2$' 676.32 H_2N-(ene)(oh)'	472.22 HN-(ene)(oh)'
3	810.37 H_2S-(oh)$_2$'		847.41 HNS-(ene)'
4	588.26 HS-(ene)(oh)'		898.43 H_3N-(oh)$_2$' 472.22 HN-(ene)(oh)'
5	898.43 H_3N-(oh)$_2$' 435.18 H_2-(oh)$_3$'		898.43 H_3N-(oh)$_2$' 449.20 H_2-(oh)$_2$'
6	898.43 H_3N-(oh)$_2$' 486.23 HN-(ene)'		847.41 HNS-(ene)' 620.29 HS-(oh)'
7	898.43 H_3N-(oh)$_2$' 662.30 H_2N-(ene)(oh)$_2$'		847.41 HNS-(ene)' 398.18 S-(ene)'
8	898.43 H_3N-(oh)$_2$' 486.23 HN-(ene)' 259.12 H-(oh)'		847.41 HNS-(ene)' 472.22 HN-(ene)(oh)'
9	898.43 H_3N-(oh)$_2$' 486.23 HN-(ene)' 268.12 N-(ene)(oh)'		847.41 HNS-(ene)' 472.22 HN-(ene)(oh)' 250.11 N-(ene)$_2$'
10	898.43 H_3N-(oh)$_2$' 486.23 HN-(ene)' 250.11 N-(ene)$_2$'		847.41 HNS-(ene)' 472.22 HN-(ene)(oh)' 245.10 H-(oh)$_2$'
11	898.43 H_3N-(oh)$_2$' 486.23 HN-(ene)' 241.11 H-(ene)'		847.41 HNS-(ene)' 472.22 HN-(ene)(oh)' 227.09 H-(ene)(oh)'
12	898.43 H_3N-(oh)$_2$' 435.18 H_2-(oh)$_3$' 245.10 H-(oh)$_2$'		898.43 H_3N-(oh)$_2$' 449.20 H_2-(oh)$_2$' 245.10 H-(oh)$_2$'
13	898.43 H_3N-(oh)$_2$' 435.18 H_2-(oh)$_3$' 213.07 H-(ene)(oh)$_2$'		898.43 H_3N-(oh)$_2$' 449.20 H_2-(oh)$_2$' 227.09 H-(ene)(oh)'
14			898.43 H_3N-(oh)$_2$' 472.22 HN-(ene)(oh)' 227.09 H-(ene)(oh)'
15			898.43 H_3N-(oh)$_2$' 472.22 HN-(ene)(oh)' 245.10 H-(oh)$_2$'
16			898.43 H_3N-(oh)$_2$' 472.22 HN-(ene)(oh)' 268.12 N-(ene)(oh)'
17			898.43 H_3N-(oh)$_2$' 472.22 HN-(ene)(oh)' 250.11 N-(ene)$_2$'

Table 26: Summarized IsoDetect output for Listing 11, where both GM1a and GM1b have been identified as expected structures. The initial ion *m/z* 1273.62 H_3NS-(oh)' is omitted from each pathway.

6.4.2. Results and Execution Times for All Studied Glycans

Having shown in detail how IsoDetect processes spectra for GM1a/GM1b, we now summarize in Table 27 results for a variety of glycans from three different biological sources.

To evaluate IsoDetect, we have elected to compare structures that have been reported in the literature against spectra collected from equivalent samples. The spectra were collected using GlySpy's Intelligent Data Acquisition; as such, no effort was made to collect spectra that would confirm or refute the expected structures.

For all tests, IsoDetect used the **`NoCrossRing`** flag to eliminate cross-ring fragment interpretations of the data. This option may discard a few structurally informative ions, but it also reduces the number of possible compositions for each observed ion, increasing throughput. Future versions of IsoDetect, which will attempt to perform linkage analysis, will of course need to find and use appropriate cross-ring fragments. For branching topology analysis, glycosidic fragments are sufficient. A relative intensity cut-off of 2% was specified.

For the IgG and ovalbumin tests, the **`-NLinkedBranching`** and **`-NLinked`** options were *not* given. Similarly, only **`-ReducingEndResidue reduced`** was specified, not the more restrictive **`-ReducingEndResidue n`** that the *N*-linked motif would demand. In both cases, these settings allow IsoDetect to call out pathways that appear to result from glycosidic cleavages but which do not fit the dogmatic *N*-linked core motif.

Source	*m/z*	Expected Structure(s)	Number of Spectra Collected	Pathways: Consistent / Total	Consistent Pathways (%)	Execution Time (seconds)
Bovine Brain Gangliosides	1273.65	GM1a: HN(S)HH-(oh) GM1b: SHNHH-(oh)	8	32/32	100%	0.49
IgG	1606.83	NH'(H')H'N'(F)n'	11	55/69	80%	0.33
	1636.84	HNH'(H')H'N'n'	7	20/45	44%	0.19
	1677.87	NH'(NH')H'N'n'	12	46/54	85%	0.40
	1810.93	HNH'(H')H'N'(F)n	8	10/59	17%	0.49
	1851.96	NH'(NH')H'N'(F)n'	9	41/41	100%	0.35
Ovalbumin	1187.61	H'(H')H'N'n'	9	25/49	51%	0.16
	1636.84	NH'(HH')H'N'n'	11	45/81	56%	0.80
	1677.87	NH'(N)(H')H'N'n'	12	52/79	66%	1.36
	1922.99	N(N)H'(N)(H')H'N'n' NH'(NH')(N)H'N'n'	14	73/89	82%	2.50

Table 27: IsoDetect results and execution times for a variety of glycans.

If the reported structures enumerated all of the glycans actually present, we would expect the Consistent Pathways column of Table 27 to register 100% in every case. We see, however, that this was the result in only two instances, GM1a/GM1b and IgG *m/z* 1851. Clearly there are unreported structures lurking in these data. We will apply GlySpy's automated abilities to these structures in Chapter 9: AUTOMATED GLYCAN TOPOLOGY ANALYSIS. As we will see, the quantity and variety of unexpected glycans makes fully-automated analysis quite difficult.

Even with these limitations, the demonstrated high performance of IsoDetect (see the Execution Time column of Table 27) enables much more efficient analysis of complex mixtures of isomeric glycans. The most time-consuming example here executed in only 2.5 seconds.

6.5. Summary

This chapter has presented the IsoDetect algorithm and experimental results. We feel that IsoDetect will prove to be a valuable tool for identifying fragments from unreported isomeric structures. But, with only OSCAR and IsoDetect available, the analyst would still be required to perform all structural analysis manually to assign these isomers. Clearly there is the need to improved tools to automatically assign these structures. Such an algorithm, called IsoSolve, is the subject of the following chapter.

CHAPTER 7:

ISOSOLVE

7.1. Overview

We have seen how OSCAR and IsoDetect improve analysts' capabilities and efficiency, but more can be done to automate the task assigning glycan topologies. In this chapter and the next, we discuss IsoSolve and Intelligent Data Acquisition (IDA), algorithms designed to operate with little or no human guidance.

IsoSolve takes as input a set of spectra (the ".raw" files generated by the LTQ) and extracts structurally-informative fragmentation pathways. IsoSolve then produces a ranked list of isomeric branching topologies that, taken together, account for the observed pathways. As structures are proposed, all pathways consistent with those structures are marked as *explained.* Intuitively we see that, if isomers are present, any single structure will explain only a fraction of the total set of fragmentation pathways. Unexplained pathways will be used to seed the search for additional isomers. As these isomers are identified, the total fraction of explained pathways will increase, until the set of proposed structures cumulatively explain all input pathways.

7.2. The IsoSolve Command

The spectra to be examined by IsoSolve are input via the AddSpectrumFile command (Section 5.2.3 on page 51). The IsoSolve command has the following format:

```
IsoSolve [NoCrossRing] MZ-target rel-intensity
```

As before, the **NoCrossRing** option specifies that compositions containing cross-ring fragments should be excluded from the analysis. The **MZ-target** parameter gives the *m/z* of the unfragmented glycan. The **rel-intensity** parameter specifies a relative intensity cutoff; peaks which fall below this threshold are ignored.

We show two example input listings for the IsoSolve command. Listing 12 shows the input for IgG *m/z* 1851.96, and Listing 13 for IgG *m/z* 1677.87. These examples will be discussed in some detail below. A 2% relative intensity cut-off has been specified.

In both cases, as with nearly every example in this and the following chapters, the spectrum files were collected via Intelligent Data Acquisition. Any spectra collected manually will be identified in the text. The goal here is to demonstrate GlySpy's ability to begin to replace the structural analyst, with the ultimate goal of performing these analyses in a completely automated, high-throughput fashion.

```
-ReducingEndResidue reduced
-NLinkedBranching

; Add all of the spectra collected via Intelligent Data Acquisition
AddSpectrumFile IGG_1851.raw
AddSpectrumFile IGG_1851_1384.raw
AddSpectrumFile IGG_1851_1384_1125.raw
AddSpectrumFile IGG_1851_1384_1125_866.raw
AddSpectrumFile IGG_1851_1384_1125_866_662.raw
AddSpectrumFile IGG_1851_1384_1125_866_662_458.raw
AddSpectrumFile IGG_1851_1592.raw
AddSpectrumFile IGG_1851_1592_1125.raw
AddSpectrumFile IGG_1851_1592_490.raw

; Run IsoSolve
IsoSolve NoCrossRing 1851.96 2
```

Listing 12: IsoSolve input for IgG *m/z* 1851.96.

```
-ReducingEndResidue reduced
-NLinkedBranching

; Add all of the spectra collected via Intelligent Data Acquisition
AddSpectrumFile IGG_1677.raw
AddSpectrumFile IGG_1677_1384.raw
AddSpectrumFile IGG_1677_1384_1125.raw
AddSpectrumFile IGG_1677_1384_1125_866.raw
AddSpectrumFile IGG_1677_1384_1125_866_662.raw
AddSpectrumFile IGG_1677_1384_1125_866_662_458.raw
AddSpectrumFile IGG_1677_1418.raw
AddSpectrumFile IGG_1677_1418_1125.raw
AddSpectrumFile IGG_1677_1418_1159.raw
AddSpectrumFile IGG_1677_1418_1159_866.raw
AddSpectrumFile IGG_1677_1418_1159_900.raw
AddSpectrumFile IGG_1677_1418_1159_900_696.raw

; Run IsoSolve
IsoSolve NoCrossRing 1677.87 2
```

Listing 13: IsoSolve input for IgG *m/z* 1677.87.

7.3. Algorithm

7.3.1. IsoSolve Goals

IsoSolve was designed to meet a variety of goals, including:

- Produce a set of isomeric topologies that together explain most or all of the observed disassembly pathways
- Produce one or more topologies per round of search, and perform additional rounds until all pathways have been explained
- Use a *seed* (a set of pathways) to begin each round
- Efficiently find the next seed to be used
- Delay generating candidate topologies until only a handful remain
- Use OSCAR's pathway constraint processing to narrow the candidate topologies

- Avoid producing the same topology more than once
- Rank the produced topologies according to the quantity and quality of supporting evidence for each structure

7.3.2. IsoSolve Overview

Figure 36 shows a high-level overview of the IsoSolve algorithm. As you can see, "interesting" seeds are generated in sequence and used to begin each search round. OSCAR's solution data structure (Section 5.3.2 on page 55) is used to convert pathways into topologies, and when a single topology is found, it is added to the ProposedStructures output set. After producing this structure, IsoSolve finds the next "interesting" seed to begin the next search round, and continues until no seeds remain.

Much of the following discussion focuses on how IsoSolve finds these "interesting" seeds and what it does with them. First, however, we cover some necessary background, including how IsoSolve delays generating structures until necessary (Section 7.3.3) and how the generated structures are scored for ranking (Section 7.3.4). Then we move on to a detailed examination of the algorithm, presented in pseudocode, and conclude with execution results.

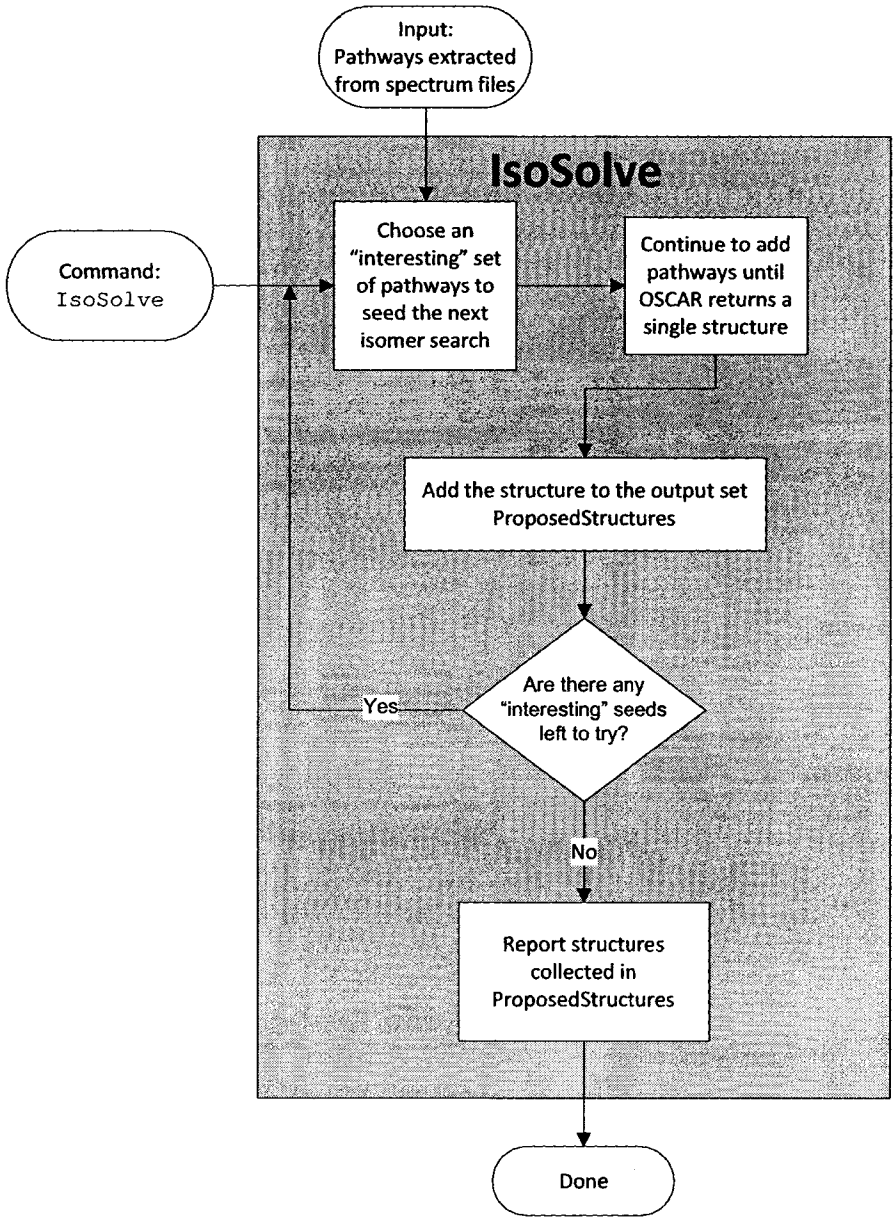


Figure 36: A very high-level overview of the IsoSolve algorithm.

7.3.3. Estimating Topology Counts for a Pathway

During its execution, IsoSolve adds sets of pathways into an OSCAR solution data structure, in effect performing an **`AddPathway`** command (Section 5.2.2) for each pathway. However, because under-constrained solutions may generate a vast number of topologies, IsoSolve delays

asking OSCAR to generate topologies from the solution until the solution can guarantee that only a handful of topologies will be produced. In effect, IsoSolve delays executing the **Summarize** command until enough pathways have been added to reduce the structure candidate set to manageable proportions.

To do this, IsoSolve calculates an upper bound on the number of branching topologies the solution can produce. To count all possible permutations of structures that this fork could produce, IsoSolve simply computes the product of the Parent Monos count for each mono. To see this, imagine that all monos in a fork have exactly one definite parent, except for some mono M1 which has three. Clearly that fork could produce only three topologies: one where M1 is connected to parent 1, one where it is connected to parent 2, and one where it is connect to parent 3. No other topologies can be generated because there is no uncertainty regarding the parent of any other mono.

Now if a second mono M2 in the fork had 2 possible parents, the fork would generate up to six topologies: each of the three topologies generated above would now have two variants, one with M2 connected to its first parent and one with M2 connected to its second parent. This logic extends to all monos in the fork, resulting in an upper bound equal to the product of the possible parent count for each mono.

Consider Table 28, which duplicates the "mono" portion of Table 12 from page 64. In this case, each of the five monos has four possible parents, and so the fork will produce no more than $4^5 = 1024$ topologies. In reality, the fork will produce fewer distinct structures, as overlapping isomorphs are eliminated. Still, taking the product of |Parent Monos| is an extremely fast way to generate an upper bound on topology count. Notice that the Children Monos and Number of Children fields of the table are ignored in this computation.

This calculation was only for one fork. To generate a similar upper bound for the entire solution, simply sum the upper bound from each fork.

Table 28: A portion of the fork from Table 12 on page 64.

The Mono Portion of Fork 0			
H^0	$H^1 H^2 N^3 n^4$	$H^1 H^2 N^3 n^4$	0 1 2 3 4
H^1	$H^0 \quad H^2 N^3 n^4$	$H^0 \quad H^2 N^3 n^4$	0 1 2 3 4
H^2	$H^0 H^1 \quad N^3 n^4$	$H^0 H^1 \quad N^3 n^4$	0 1 2 3 4
N^3	$H^0 H^1 H^2 \quad n^4$	$H^0 H^1 H^2 \quad n^4$	0 1 2 3 4
n^4	$H^0 H^1 H^2 N^3$	$H^0 H^1 H^2 N^3$	0 1 2 3 4

Table 29 shows a portion of the same fork after many more inference rules have successfully been applied. Now the upper bound is 2x2x2x1 = 8. This follows because H^0, H^1, and H^2 each have 2 possible parents (H^1/N^3, H^0/N^3, and H^0/H^1, respectively, and so 2x2x2) and N^3 has one possible parent (n^4, and so x1). The mono n^4 has no parent because it is the root, but we obviously do not multiply a zero into our upper bound, and so the fork can produce no more than 8 distinct branching topologies. Again notice that only the Parent Monos column is used in this calculation; the Children Monos and Number of Children columns are ignored[10].

[10] Eagle-eyed readers may notice a seeming inconsistency in Table 29: H^1 has H^0 as a possible parent, but H^0 does not have H^1 as a possible child. If this asymmetry seems galling, recall that OSCAR is not done processing this fork. Additional inference rules will be applied to eliminate H^0 as H^1's parent. However, this is all irrelevant to the matter at hand: efficiently estimating topology counts for a fork.

Table 29: A portion of the fork from Table 17 on page 74.

The Mono Portion of Fork 0			
H^0	H^1 N^3		0
H^1	H^0 N^3	H^0 H^2	2
H^2	H^0 H^1		0
N^3	$\boxed{n^4}$	H^0 H^1	1
n^4		$\boxed{N^3}$	1

Clearly as more inference rules are applied and the Parent Monos column is reduced further, the topology upper bound will approach one. It is through these estimates that IsoSolve measures its progress: as it tentatively adds a pathway to the solution, it can determine if the pathway helped the solution converge toward a single topology. If it did, the pathway is retained; if not, the pathway is discarded. IsoSolve adds all pathways in this fashion, one by one, until either a single topology is found or all pathways have been tried. In either case, structures are not generated until very late in the search. This is an essential design detail that leads to improved performance.

7.3.4. Rank Scoring

The isomers produced by IsoSolve are scored and ranked in order to provide the analyst with additional information about how well each structure fits with the accumulated data.

As each isomer is generated, its consistency is checked with every input pathway. So if there are 100 pathways, each isomer has a computed subset of consistent pathways. (This computation is done using GlySpy's IsoDetect technology.) These pathways form the basis of the structure scoring.

A simple scoring strategy was implemented first, where a simple count of consistent pathways was performed. If structure S1 was consistent with 60 pathways, its score was 60/100 = 60%. This was quickly seen to be unacceptable. In this case, S1 is *not* consistent with 40 pathways—perhaps because those 40 pathways reside on spectra that could not possibly have come from structure S1. Because MS^n allows for the separation of isomers directly in the mass spectrometer, we should not penalize S1 simply because we collected 40 pathways from a different isomer. An example may help.

When processing the GM1a/GM1b, one input spectrum has the pathway 1273.6_847.4 (Spectrum A-13 on page 214). Three pathways terminate on this spectrum: 1273.6_847.4_398.1, 1273.6_847.4_472.2, and 1273.6_847.4_620.3. However, *none* of these are expected to be consistent with GM1a because 1273.6_847.4, the *spectrum's* disassembly pathway, is not consistent with GM1a. Ion *m/z* 847.4 has composition HNS-(ene)' and could only have come from GM1b, not GM1a. Clearly we should not penalize GM1a's score because we collected spectra specific to GM1b!

IsoSolve's final implemented scoring scheme addresses these problems. Each structure's final score is ConsistentPathways/AvailablePathways, expressed as a percentage, where:

- ConsistentPathways is the number of pathways consistent with the structure, and
- AvailablePathways is the total number of pathways that terminate on consistent spectra.

In the GM1a/GM1b example, GM1a is consistent with 15 pathways and a total of 20 pathways terminate on spectra consistent with GM1a. Therefore, GM1a's final score is 15/20 = 75%. Similarly, GM1b's score is 19/26 = 73%.

7.3.5. IsoSolve Pseudocode

With that necessary background behind us, we now move on to a detailed discussion of the IsoSolve algorithm as implemented by three procedures: DoIsoSolve, DoIsoSolveForSeed, and ProposeStructuresFromSeed. We also briefly discuss IsoSolve's input and variables.

Roughly speaking, the three procedures fulfill these roles:

- DoIsoSolve: Guarantees that every pathway has the chance to be a seed
- DoIsoSolveForSeed: Generate the next interesting seed
- ProposeStructuresFromSeed: Propose at least one well-supported structure consistent with the given seed

7.3.5.1 The AllPathways and ProposedStructures Variables

Listing 14 defines two variables used by IsoSolve: AllPathways and ProposedStructures.

- INPUT: AllPathways is the set of disassembly pathways to be examined
- OUTPUT: ProposedStructures collects the topologies proposed by IsoSolve; these topologies are presented to the user when the algorithm concludes.

These variables will be referenced in the pseudocode and flowcharts that follow. Although these variables are presented as if they are globals, they are in fact encapsulated in a proper C++ class. Similar liberties have been taken below in an attempt to simplify implementation details into more presentable pseudocode.

```
01:   // INPUT: The set AllPathways contains all pathways to be examined
02:   Variable AllPathways has type Set of Pathways;
03:
04:   // OUTPUT: The set ProposedStructures collects topologies as
05:   // IsoSolve proposes them.
06:   Variable ProposedStructures has type Set of Topologies;
```

Listing 14: The variables AllPathways and ProposedStructures as used by IsoSolve. AllPathways is the input set of disassembly pathways to be considered; ProposedStructures is the output set of proposed glycan topologies.

A Note on Line Numbers

The listings in this chapter (Listing 14 through Listing 17) are shown with line numbers (01-06 in the listing above). These line numbers are intentionally continued from one listing to the next, to minimize confusion when the following text refers to, say, line 20.

7.3.5.2 The DoIsoSolve Procedure

The DoIsoSolve procedure guarantees that every pathway has the chance to be a seed. It is illustrated in Figure 37 and described in pseudocode in Listing 15. This procedure loops over all of the input pathways and, one by one, uses each as the seed parameter to DoIsoSolveForSeed. The necessity of this will become clearer as our discussion continues, but intuitively we should recognize that every pathway has an equal opportunity to be a seed that generates matching structures. We do not bias the structures generated by the order in which pathways are selected as seeds.

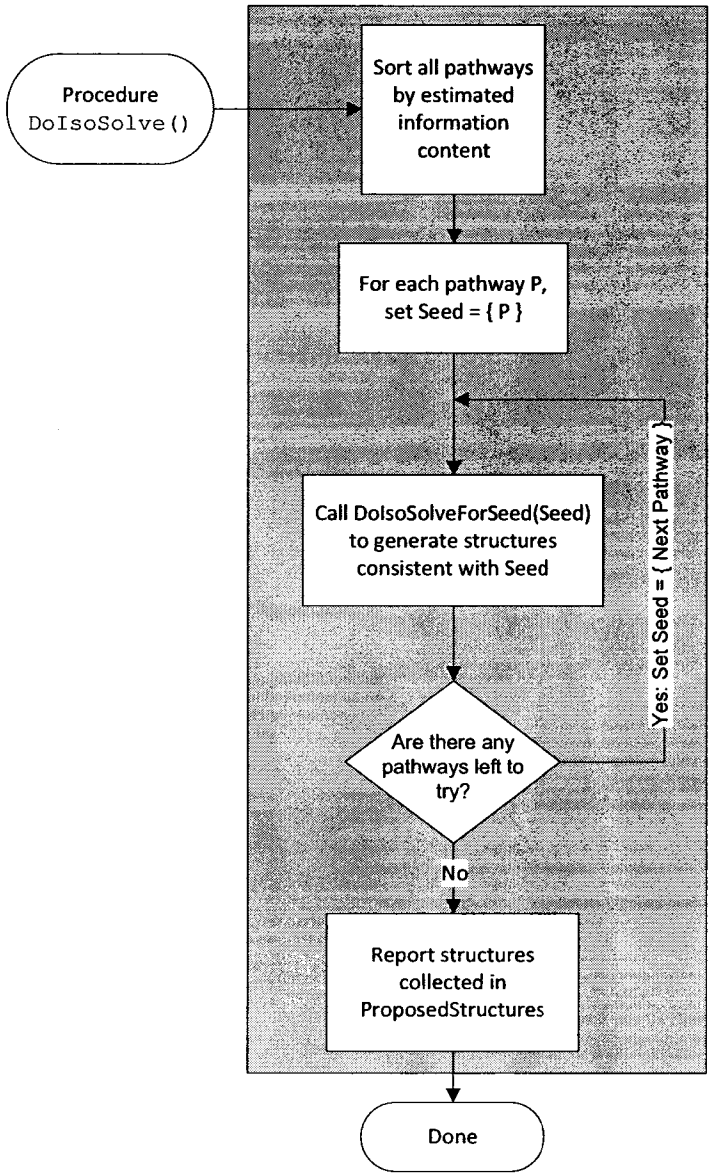


Figure 37: A flowchart for the procedure DoIsoSolve.

```
07:   Procedure DoIsoSolve()
08:   {
09:       // Initially, no structures have been proposed, so
10:       // the ProposedStructures set is empty.
11:       ProposedStructures = { };
12:
13:       // The variable AvailableSeeds is the set of pathways available
14:       // to be added to the seed.  Initially, AvailableSeeds contains
15:       // all pathways, but it shrinks over time.
16:       Variable AvailableSeeds has type Set of Pathways;
17:       AvailableSeeds = AllPathways;
18:
19:       // Every pathway gets to be the starting seed of a new search.
20:       for each pathway P in AllPathways do {
21:           // The variable Seed is the set of pathways used as
22:           // the starting point for the structure search.
23:           // Give pathway P its chance to be the seed.
24:           Variable Seed has type Set of Pathways;
25:           Seed = { P };
26:
27:           // Remove pathway P from the available seeds.  P is already
28:           // in Seed, so it is no longer available to be added.
29:           Remove P from AvailableSeeds;
30:
31:           // Run a search starting with this seed.  This will
32:           // add to the ProposedStructures set if successful.
33:           DoIsoSolveForSeed(Seed, AvailableSeeds);
34:       }
35:
36:       // The output includes the structures found, the pathways
37:       // and spectra consistent with each structure, and the pathways
38:       // that were required to generate the structure.
39:       Report ProposedStructures to the user;
40:   }
```

Listing 15: Procedure DoIsoSolve implements the top-level IsoSolve processing. Proposed topologies are gathered in the variable ProposedStructures and reported to the user.

However, do not think that a seed must be a single pathway. As we will see, seeds are more commonly a *set* of pathways.

In fact, perhaps the key observation in understanding IsoSolve is this: *A seed is a collection of pathways for which no consistent structure has yet been proposed.* (A structure is consistent with a seed if it is consistent with all of the pathways in the seed.) So, *IsoSolve's main task is to*

generate a small number of seeds and then to propose a small number of topologies for each seed.

However, generating the seeds is not a trivial undertaking. If there are N pathways, there are 2^N-1 possible non-empty subsets of pathways. For the examples in this document, N ranges from 32 to 89, yielding astronomical numbers of possible seeds: 2^{89}-1 is 618,970,019,642,690,137,449,562,111. To deal with this explosion, the next procedure discussed, DoIsoSolveForSeed, recognizes when a seed is consistent with a previously- proposed structure.

To improve pathway selection, IsoSolve sorts the available pathways in order of information content, the goal being to move the most structurally informative pathways to the beginning of the list, so IsoSolve can reduce the number of candidate structures quickly. For each pathway, IsoSolve computes an upper bound for the number of topologies that pathway alone could generate (Section 7.3.3). It then sorts the pathways in increasing order, so that the pathways that generate the fewest structures are processed first.

7.3.5.3 The DoIsoSolveForSeed Procedure

The DoIsoSolveForSeed procedure generates interesting seeds for which topologies should be produced. This procedure is detailed in Figure 38 and Listing 16.

This procedure examines the given seed, generates consistent structures from it (if necessary), and then removes the appropriate pathways from the AvailableSeeds set. The AvailableSeeds variable tracks those pathways that might profitably be combined with the current seed to form a new seed for the next search round. If the current seed contains N pathways, the next seed generated will contain N+1. This seed augmentation is done in such a way as to guarantee that each successive seed cannot produce a previously-proposed structure.

As such, IsoSolve quickly finds "interesting" seeds worthy of further consideration without ever wasting time generating duplicate structures.

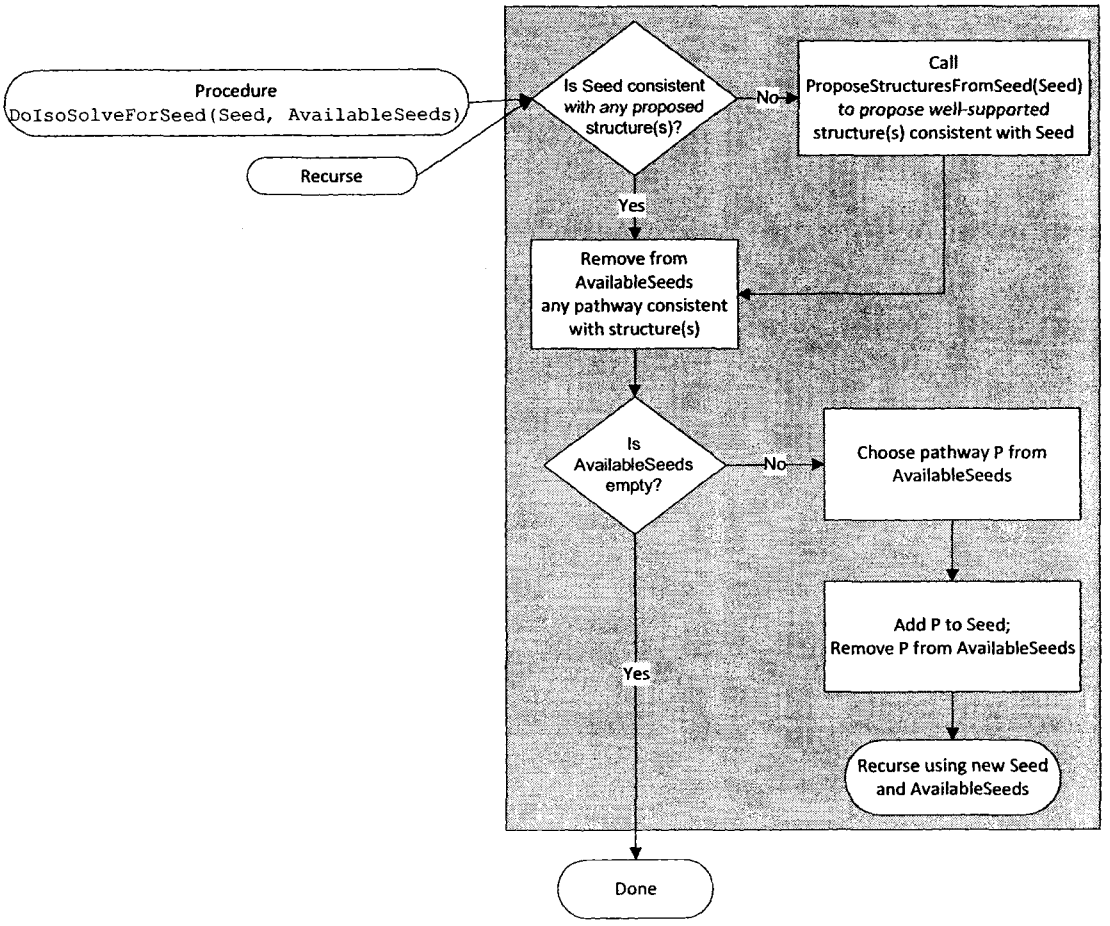


Figure 38: A flowchart for the procedure DoIsoSolveForSeed(Seed, AvailableSeeds).

```
41:   Procedure DoIsoSolveForSeed(Seed, AvailableSeeds)
42:       // Seed is the set of pathways used to start the search.
43:       Parameter Seed has type Set of Pathways;
44:
45:       // AvailableSeeds is the set of pathways available
46:       // to be added to the seed.
47:       Parameter AvailableSeeds has type Set of Pathways;
48:   {
49:       if (Seed is consistent with at least one proposed structure) {
50:           // There is no reason to generate new structures for this
51:           // Seed since we already have structures compatible with
52:           // Seed.  Instead, remove any pathway consistent with those
53:           // structures from AvailableSeeds, since those pathways
54:           // also no longer need to be explained.
55:           for each proposed structure S that is consistent with Seed
56:               Remove S.ConsistentPathways from AvailableSeeds;
57:       } else {
58:           // Seed is not consistent with any proposed structure.
59:           // Use this Seed to generate one or more Structures that are
60:           // well-supported by the full pathway set, not just the
61:           // remaining AvailableSeeds.
62:           NewStructures = ProposeStructuresFromSeed(Seed);
63:
64:           // The structure(s) just proposed will be consistent with a
65:           // number of pathways.  Remove those pathways from
66:           // AvailableSeeds, since they no longer need to be explained.
67:           // Also, add each structure to the ProposedStructures set.
68:           for each structure S in NewStructures do {
69:               Add S to ProposedStructures;
70:               Remove S.ConsistentPathways from AvailableSeeds;
71:           }
72:       }
73:
74:       // If AvailableSeeds is an empty set, search is completed.
75:       if (AvailableSeeds is not empty) {
76:           // Some available seeds are still not consistent with any
77:           // structure generated from this seed.  Select one still-
78:           // unexplained pathway P, add it to Seed and remove it from
79:           // AvailableSeeds.
80:           P = Choose pathway from AvailableSeeds;
81:           Add P to Seed;
82:           Remove P from AvailableSeeds;
83:
84:           // Recurse to start a new search using the augmented seed.
85:           DoIsoSolveForSeed(Seed, AvailableSeeds);
86:       }
87:   }
```

Listing 16: Procedure DoIsoSolveForSeed manages the search for structures consistent with a given seed of pathways. It finds well-supported structures consistent with Seed and consumes the AvailableSeeds set until it becomes empty.

As you can see, the first action taken by DoIsoSolveForSeed is to determine if the given seed is consistent with any structure that has already been proposed. If it is, the algorithm examines these structures and removes their consistent pathways from the AvailableSeeds set. So, if the current seed is consistent with proposed structure S1, and S1 is consistent with pathways P1, P5, and P6, the algorithm removes P1, P5, and P6 from AvailablePathways. Intuitively, this means that there is no need to combine the current seed with pathway P1, P5 or P6 at any time in the future. If we did, the seed would just lead us back to structure S1!

However, if the seed is not compatible with any proposed structure, the procedure calls ProposeStructuresFromSeed, discussed next. That function returns one or more structures consistent with the given seed. As above, we examine each of these newly generated structures and remove their compatible pathways from AvailablePathways. This, again, is to prevent us from considering a future seed that leads back to previously-proposed structures.

After this initial structure lookup or creation, DoIsoSolveForSeed must decide if there are any interesting seeds left to be examined. If AvailableSeeds is empty at this point, there are no useful ways to extend the seed, and the procedure is done.

However, consider the case when AvailableSeeds is *not* empty. For example, suppose AvailableSeeds contains just one pathway, P9, and seed contains P4 and P5. This means that no structure has yet been proposed that is consistent with all three pathways, P4, P5, and P9. Obviously, we should attempt to generate a structure consistent with all three of these pathways, and so we extend seed from { P4, P5 } to { P4, P5, P9 } and run DoIsoSolveForSeed again with this new seed. In this way we have quickly identified an interesting seed by merely observing what single-pathway augmentation of the seed has not yet yielded a proposed structure.

Also note that we remove great swaths of pathways from AvailablePathways each time a structure is proposed—namely, every pathway consistent with that new structure. In practice, the AvailablePathways set shrinks very rapidly, allowing the algorithm to achieve a high performance regardless of the 2^N-1 possible seeds discussed in Section 7.3.5.2.

7.3.5.4 The ProposeStructuresFromSeed Function

The ProposeStructuresFromSeed function proposes at least one well-supported structure consistent with the seed it is given. See Figure 39 and Listing 17.

The function performs this task in a relatively straightforward manner. Using the topology count estimation technique of Section 7.3.3, it tracks the number of topologies that an OSCAR solution could generate for a given set of pathways. It then tentatively adds another pathway from the full data set and asks if the solution now generates fewer structures. If yes, the pathway is kept as part of the solution; if no, the pathway is discarded. In either case, all pathways in the full data set are tentatively added in this fashion, until the solution generates a single topology. The function then exits, returning the topology to the caller.

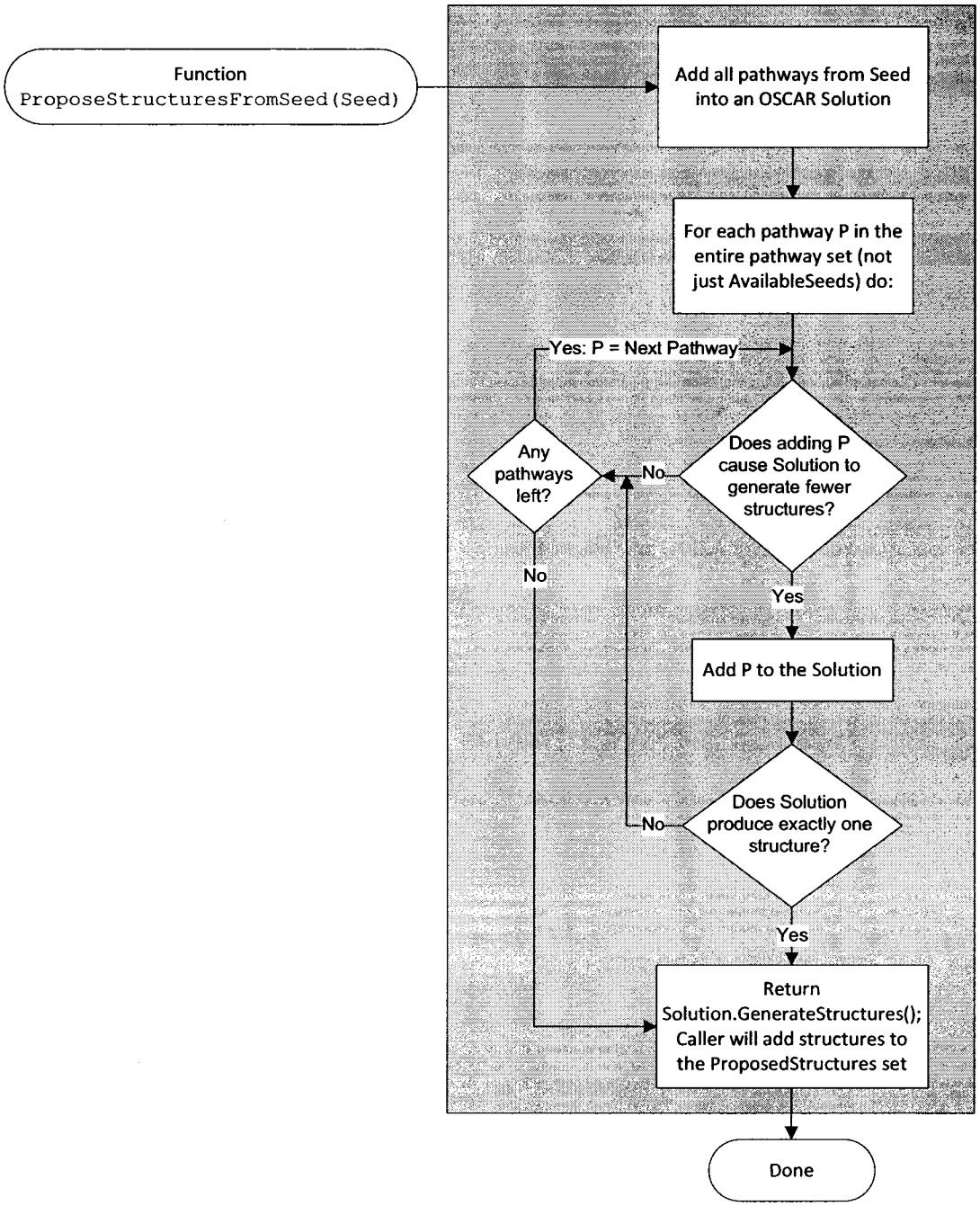


Figure 39: A flowchart for the function ProposeStructuresFromSeed(Seed).

```
88:   Function ProposeStructuresFromSeed(Seed)
89:       Parameter Seed has type Set of Pathways;
90:
91:       Returns Set of Structures;
92:   {
93:       // The variable Solution is implemented by
94:       // OSCAR's "Solution" data type
95:       Variable Solution has type OSCAR Solution;
96:
97:       // Execute "AddPathway P" for each pathway P in Seed
98:       Add all pathways from Seed to Solution;
99:
100:      // If the seed cannot produce any structures, abort
101:      // the search: Return the empty set and exit the function.
102:      if (Solution is sterile)
103:          return { };
104:
105:      // Search all pathways in order, looking for ones that help
106:      // Solution converge toward a single structure
107:      for each pathway P in AllPathways do {
108:          if (adding P to Solution makes progress) {
109:              Add P to Solution;
110:
111:              // If we have converged on a single topology, return it
112:              // and exit the function.
113:              if (Solution generates exactly 1 structure)
114:                  return Solution.GenerateStructures();
115:      }
116:
117:      // The algorithm did not converge on a single structure, so
118:      // return the set of all structures that Solution can generate.
119:      return Solution.GenerateStructures();
120:  }
```

Listing 17: The ProposeStructuresFromSeed function returns a set of well-supported structures that are consistent with the given set of Seed pathways. It tentatively combines Seed with all pathways in IsoSolve's data set (not just the AvailableSeeds) to converge toward a small number of proposed topologies.

A few clarifications are in order here.

First, because ProposeStructuresFromSeed tentatively adds every available pathway, there is no bias against selecting pathways that are consistent with some previously-proposed structure. We know at this point that the seed is guaranteed to lead to new structures, and so duplicate structures are not an issue. However, we cannot limit this tentative pathway selection

to some subset of the pathways, or we may fail to generate a valid structure. Just because pathway P1 is part of structure S1 doesn't mean it can't also be part of structure S2. For example, recall from the GM1a/GM1b example (Table 26 on page 108) that some pathways can be compatible with multiple isomers. In that case, both 1273.6_898.4 and 1273.6_989.4_676.3 are compatible with GM1a and GM1b. When searching for GM1b, it would be a mistake to ignore pathway 1273.6_898.4 just because the pathway is consistent with GM1a.

Second, it is possible that an isomer is present but there are insufficient data for ProposeStructuresFromSeed to find a combination of pathways that yields exactly and only that structure. In this case, the OSCAR solution manipulated by the function would fail to converge on a single structure before running out of pathways to tentatively add. Here, ProposeStructuresFromSeed will return multiple structures—namely, all of the structures produced by the solution. These structures are marked with a count of structures produced in this batch, and so the analyst can easily identify the structures that were produced as a "bunch" due to insufficient data. A skilled analyst could then determine which further spectra to collect to resolve the ambiguity.

7.4. Limitations/Future Work

From the above discussion, we can see a number of areas where IsoSolve can be improved in future efforts:

1. ProposeStructuresFromSeed could be modified to perform a more intelligent selection of the next pathway to tentatively add to the OSCAR solution. Currently the selection is made based on the estimated information content of the pathways (as estimated by the technique described in Section 7.3.3 on page 116), but other selection strategies should be considered. For example, selecting a complementary

ion may be profitable, as we saw when sequencing the fetuin glycan *m/z* 3618.8 (Section 5.5 on page 84).

2. ProposeStructuresFromSeed uses OSCAR to calculate an upper bound on the number of structures the solution might generate, but that upper bound can sometimes be quite a bit higher than the actual number. This could lead IsoSolve to discard a pathway that actually reduced the number of candidate structures. Better techniques for quickly estimating topology counts would be beneficial here.

3. The scoring algorithm used to rank proposed structures suffers slightly in the presence of isobars, as discussed in Section 7.3.4. One improvement would be to avoid penalizing scores if multiple fragments on one spectrum are inconsistent with each other. That is, inconsistent pathways wouldn't all count toward AvailablePathways; rather, only the largest subset of consistent pathways would be counted.

7.5. Results and Discussion

We now demonstrate how IsoSolve processes two tests cases from IgG, *m/z* 1851.96 and *m/z* 1677.8. Line numbers called out in the following discussion refer to the pseudocode of Listing 14 through Listing 17.

7.5.1. IgG *m/z* 1851.96

Intelligent Data Acquisition was used to collect nine spectra for IgG glycan *m/z* 1851.96, from which IsoSolve extracted 41 pathways. These pathways were then sorted by estimated information content. A portion of this sorted pathway list is shown in Table 30.

#	Pathway	Estimated Topologies
P1	1851.96_1384.68_1125.55_921.37	3
P2	1851.96_1384.68_1125.55_866.40_662.30_458.27_268.03	4
P3	1851.96_1384.68_1125.55_866.40_662.30_458.27_212.91	4
P4	1851.96_1384.68_1125.55_866.40_662.30_458.13	4
P5	1851.96_1384.68_1125.55_866.40_458.15	4
P6	1851.96_1384.68_1125.55_866.40_662.30_417.07	9
P7	1851.96_1384.68_1125.55_866.40_662.30_268.01	9
P8	1851.96_1384.68_1125.55_866.40_662.30_435.12	9
P9	1851.96_1384.68_1125.55_866.40_639.21	9
P10	1851.96_1384.68_1125.55_866.40_621.20	9
P11	1851.96_1384.68_921.35	12
P12	1851.96_1384.68_1125.55_866.40_662.23	16
P13	1851.96_1384.68_1125.55_866.40_417.08	16
P14	1851.96_1384.68_1125.55_866.40_435.22	16
P15	1851.96_1384.68_1125.55_866.33	16
P16	1851.96_1384.68_1125.55_898.35	16
P17	1851.96_1384.68_866.32	16
P18	1851.96_1592.70_1125.40_921.34	18
P19	1851.96_1592.70_921.32	18
P20	1851.96_1592.70_490.16_302.02	20
...	...	...

Table 30: The first 20 pathways extracted from the spectrum files for IgG glycan *m/z* 1851.96.

Now DoIsoSolve begins iterating over the pathways (line 20), trying each as a seed. The first seed is a set containing the sole pathway { P1 } , and is passed to DoIsoSolveForSeed (line 33).

Since no structures have yet been proposed, we execute line 62, calling ProposeStructuresFromSeed with seed = { P1 }.

Pathway P1 is estimated to produce up to three structures. Because this count is small, ProposeStructuresFromSeed actually generates all possible structures, and three structures are created. DoIsoSolveForSeed then iterates over all of the pathways (line 107), adding each in turn to the seed, { P1 }. The pathway P2 is tentatively added to the seed, and { P1, P2 } is found to generate exactly one structure: NH'(NH')H'N'(F)n', designated S1. DoIsoSolveForSeed returns S1 to the caller (line 114).

Back in DoIsoSolveForSeed (line 69), S1 is added to the ProposedStructures set. Next, the pathways consistent with this structure are removed from AvailableSeeds. But *every* pathway is consistent with S1, and so AvailableSeeds becomes empty, and the function exits.

Returning to DoIsoSolve (line 33), the algorithm loops (line 20) and passes { P2 } as the seed for the next invocation of DoIsoSolveForSeed (line 33 again). This time, however, the condition on line 49 is met and we do not generate a structure for the seed { P2 }. Instead, we examine the previously-generated structure (S1) and remove all of its compatible pathways from AvailableSeeds, which again becomes empty, and again causes DoIsoSolveForSeed to exit.

The loop in DoIsoSolve (line 20), continues, trying seeds { P3 }, { P4 }, ..., { P41 }, but in each case DoIsoSolveForSeed finds that structure S1 is compatible with the seed and so no further processing is done.

For this example, only a single structure is proposed, and the minimum possible number of seeds (41) has been considered. The algorithm has clearly performed very well for this case, where isomers do not appear to be present. Next we look, more succinctly, at IgG glycan *m/z* 1677.8, where two isomers are present.

7.5.2. IgG *m/z* 1677.8

For IgG glycan *m/z* 1677.8, a total of 12 spectra were collected by Intelligent Data Acquisition, from which 47 pathways were extracted. The first 15 of these are shown in Table 31.

Table 32 gives an abridged representation of how IsoSolve processes these pathways. The Context column provides some indication of what the algorithm is doing, and the Action/Result column describes what is accomplished at this step. The "Seed = { Pn }" context entries represent the loop in DoIsoSolve, where each pathway is given a chance to be its own seed. Indented entries, like "Try { P1, P2 }" and "Recurse with seed = { P1, P13 }" represent the processing in DoIsoSolveForSeed.

#	Pathway	Estimated Topologies
P1	1677.87_1384.64_1125.54_921.34	3
P2	1677.87_1418.60_1125.50_921.44	3
P3	1677.87_1384.64_1125.54_866.40_662.36_458.20_268.11	4
P4	1677.87_1384.64_1125.54_866.40_662.36_458.13	4
P5	1677.87_1384.64_1125.54_866.40_662.36_245.04	4
P6	1677.87_1384.64_1125.54_866.40_662.36_226.81	4
P7	1677.87_1384.64_1125.54_866.40_458.26	4
P8	1677.87_1418.60_1214.55	4
P9	1677.87_1214.49	4
P10	1677.87_1384.64_1125.54_866.40_662.36_417.16	9
P11	1677.87_1384.64_1125.54_866.40_662.36_268.13	9
P12	1677.87_1384.64_1125.54_866.40_662.36_435.25	9
P13	1677.87_1384.64_1125.54_866.40_662.36_444.11	9
P14	1677.87_1418.60_1159.50_866.36_639.31	9
P15	1677.87_1384.64_1125.54_866.40_639.31	9
...	...	...

Table 31: The first 15 pathways extracted from the spectrum files for IgG glycan *m/z* 1677.8.

Context	Action/Result
Seed = { P1 }	Generates 3 structures, so try adding more pathways
Try { P1, P2 }	Still generates 3 structures, so discard P2
Try { P1, P3 }	Generates one structure, **NH'(NH')H'N'n', designated S1**
	Structure S1 is consistent with every pathway except P13, so AvailableSeeds shrinks to { P13 }
Recurse with seed = { P1, P13 }	Generates 2 structures, so try adding more pathways. None help converge until P10.
Try { P1, P10, P13 }	Generates one structure, **NH'(N)(H')H'N'n', designated S2.** S1 and S2 combined are consistent with all input pathways.
	AvailableSeeds becomes empty
Seed = { P2 }	P2 is consistent with S1 and S2, so AvailableSeeds becomes empty.
Seed = { P3 }	P3 is consistent with only S1, so AvailableSeeds becomes { P13 }
Recurse with seed = { P3, P13 }	Generates no topologies (P3 and P13 are inconsistent with each other), so exit
Seed = { P4 }	P3 is consistent with only S1, so AvailableSeeds becomes { P13 }
Recurse with seed = { P4, P13 }	P4 and P13 are inconsistent, so exit
Seed = { P5 }	P5 is consistent with only S1, so AvailableSeeds becomes { P13 }
Recurse with seed = { P5, P13 }	P5 and P13 are inconsistent, so exit
...	...
Seed = { P8 }	P8 is consistent with S1 and S2, so AvailableSeeds becomes empty.
Seed = { P9 }	P9 is consistent with S1 and S2, so AvailableSeeds becomes empty.
...	...
Seed = { P47 }	P47 is consistent with S1 and S2, so AvailableSeeds becomes empty.

Table 32: A compressed representation of IsoSolve's execution over the pathway data set for IgG glycan *m/z* 1677.8.

Note how quickly IsoSolve discovered structures S1 and S2. S1 is generated after trying only three seeds, { P1 }, { P1, P2 }, and { P1, P3 }; S2 is generated after just two more seeds, { P1, P13 } and { P1, P10, P13 }.

Once S1 and S2 are proposed, the algorithm searches for seeds that are inconsistent with these structures. For example, { P2 } is discarded because it is consistent with S1 and S2, which are themselves consistent with all input pathways. This indicates that P2, by itself, is not guaranteed to lead to a structure that has not already been proposed, and is discarded.

Seed { P3 } is tried next. Because P3 is consistent with S1, and S1 is consistent with every pathway except P13, the only seed worth considering is { P3, P13 }. That is, this seed is the only one that includes P3 and could possibly generate a new structure. However, this pair of pathways is internally inconsistent—they generate no structures when given to OSCAR—and is also discarded.

The pattern occurs through the rest of the outer loop, where some pathways such as P4 and P5 are combined with P13 to yield sterile seeds (seeds that generate no structures), and other seeds, such as P8 and P9, are consistent with S1 and S2 and therefore have no unexplained pathways with which to be combined.

An enterprising analyst might at this point wonder why pathway P13 is called back several times. This pathway, *m/z* 1677.87_1384.64_1125.54_866.40_662.36_444.11, represents disassembly down to an *N*-linked core that contains a bisecting HexNAc. This contrasts with pathways that pass through ion *m/z* 458.2 (such as P3 and P4). This ion represents the *N*-linked core motif without a bisecting HexNAc. Clearly ions *m/z* 444.1 and 458.2 must come from structural isomers, and IsoSolve has automatically focused its attention on these ions.

Listing 18 shows some of IsoSolve's output for this case. We see that two topologies have been proposed (NH'(NH')H'N'n' and NH'(N)(H')H'N'n'), and we see the calculated score for each (97.87% and 91.30%, respectively). We are shown which seeds were used in both cases (<1> and <1,13>, which represent { P1 } and { P1, P13 }, respectively), and the exact pathways that combined to produce the structures (<1,3> and <1,10,13>). Importantly, IsoSolve also presents the complete list of pathways that are consistent with each structure, allowing the analyst to make further judgments about the quality of support for each.

```
IsoSolve results:
Structures found: 2
--------- Structure 1 of 2 ---------
Linear code (branching): NH'(NH')H'N'n'
FINAL SCORE: 97.87
Order found: 1
Seed Pathways: <1>
Generated from these Pathways: <1,3>

Compatible Pathways:
46 compatible out of 47 possible:  97.87%
46 compatible out of 47 total:     97.87%
<1,2,3,4,5,6,7,8,9,10,11,12,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29
,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47>
-------------------------------------
--------- Structure 2 of 2 ---------
Linear code (branching): NH'(N)(H')H'N'n'
FINAL SCORE: 91.30
Order found: 2
Seed Pathways: <1,13>
Generated from these Pathways: <1,10,13>

Compatible Pathways:
42 compatible out of 46 possible:  91.30%
42 compatible out of 47 total:     89.36%
<1,2,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,
32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47>
-------------------------------------
```

Listing 18: Abridged IsoSolve output for IgG glycan *m/z* 1677.8.

7.6. Validation

IsoSolve was applied to a variety of glycan samples whose spectra were collected by use of the Intelligent Data Acquisition module. See Chapter 9: AUTOMATED GLYCAN TOPOLOGY ANALYSIS for a discussion of these results along with their execution times.

CHAPTER 8:

INTELLIGENT DATA ACQUISITION (IDA)

8.1. Overview

Higher-order MS^n analysis brings challenges that have hindered its adoption as the premier glycoanalytic methodology. Of course, the overriding problem has been the lack of analytical tools and techniques to interpret the collected spectra. These issues are the focus of the GlySpy algorithms OSCAR, IsoDetect, and IsoSolve.

However, another significant drawback is the lack of tools to *collect* these spectra automatically. When an analyst is confronted with spectra generated from a glycan, the obvious question is, "Which ion should be fragmented next?" This is the question that GlySpy's Intelligent Data Acquisition (IDA) module attempts to answer. Given a set of spectra input by the **`AddSpectrumFile`** command (Section 5.2.3), IDA's **`SuggestPeaks`** command provides several inquiry modes to suggest ions worthy of further fragmentation.

Many mass spectrometers are capable of automated data acquisition, where the analyst typically defines masses and neutral losses of interest, and the instrument dutifully collects sets of mass spectra for ions that meet these constraints. However, these capabilities are currently quite limited and often result in the collection of many redundant or useless spectra. Even the best commercially-available data acquisition software, for example Thermo Fisher Scientific's,

remains inadequate for oligosaccharide analysis, often collecting redundant or uninformative spectra (Ashline 8).

GlySpy's Intelligent Data Acquisition module attempts to address these shortcomings, effectively replacing the analyst as the director of data acquisition.

8.2. The SuggestPeaks Command

GlySpy exposes IDA's capabilities via the **SuggestPeaks** command, which has this form:

```
SuggestPeaks [NoPrune] Mode RelInt AbsInt SortOrder pathway
```

Mode defines the desired collection mode, that is, the type of fragment search the analyst wishes to perform, and is detailed in Table 33. **RelInt** specifies a minimum relative intensity as a percentage; the command will not return peaks below this intensity. **AbsInt** specifies an analogous absolute intensity cut-off value, where the absolute intensity is approximated by the product of the peak's relative intensity and its spectrum's normalization level. **SortOrder** specifies the sort order of the suggested peaks (Table 34). The **pathway** parameter limits the search's scope to spectra that have an equivalent disassembly pathway prefix. Finally, the **NoPrune** option, if given, allows the command to return peaks that are likely to be structurally redundant with spectra that have already been collected; the default behavior, without the option, will prune these ostensibly redundant peaks to avoid swamping the analyst with spectra of questionable value.

Table 33: The Mode parameter for the SuggestPeaks command.

Mode	Returned Peaks	Intended Use
Auto	Variety of peaks useful for making structural assignments.	Automate data collection. See Section 9.2.3.
MajorGlyco	Highest intensity peaks that could have resulted strictly from glycosidic cleavages.	Traverse deepest MS^n pathway. See Section 9.2.1.
MissingComplements	Peaks that are complementary to an existing spectrum's pathway.	Identify and isolate lost complementary fragments. See Section 9.2.2.
IsoDetect	Peaks that IsoDetect has flagged as indicating possible isomers.	Collect spectra that likely come from isomeric structures. See Section 9.2.4.
OnlyReducingEndScarred	Peaks that have a composition where only the reducing end is scarred.	Isolate all losses on reducing end of fragment.
OnlyNonReducingEndScarred	Peaks that have a composition where only the non-reducing end is scarred.	Isolate all losses on non-reducing end of fragment.
SingleSideScarred	Peaks that have a composition where only the reducing end or the non-reducing end is scarred, but not both.	A combination of OnlyReducingEndScarred and OnlyNonReducingEndScarred.
EneScar	Peaks that have a composition containing at least one (ene)-type scar.	Isolate B-type ions likely to produce structurally informative cross-ring fragments.

Table 34: The `SortOrder` parameter for the SuggestPeaks command.

SortOrder	Sort Criterion	Sort Order
RankByAbsInt	Absolute intensity	Decreasing
RankByRelInt	Relative intensity	Decreasing
RankByEstTopologies	Number of estimated topologies compatible with this pathway	Increasing
RankByMSnDepth	MS^n depth	Decreasing

Listing **19** shows several possible uses of the **SuggestPeaks** command. Notice that the **Mode** and **SortOrder** parameters can be used in any combination. The **Pathway** parameter can specify either a full, unfragmented glycan (single *m/z*) or a pathway (e.g., 1677.87_1125.54_662.5), and serves to restrict the scope of the search. In all cases shown, the relative intensity cut-off is given as 5% and the absolute intensity cut-off is 100.

```
SuggestPeaks MajorGlyco          5 100 RankByRelInt          1677.87
SuggestPeaks IsoDetect           5 100 RankByEstTopologies   1677.87_1384.5
SuggestPeaks MissingComplements  5 100 RankByAbsInt          1677.87_1125.54_662.5
SuggestPeaks EneScar             5 100 RankByMSnDepth        1677.87_1384.5
```

Listing 19: A few possible SuggestPeaks commands, showing a mixture of collection modes and sort orders.

The **SuggestPeaks** command refrains from suggesting peaks that already have an associated spectrum. For example, if the spectrum for *m/z* 1677.9 contains an interesting peak at *m/z* 1384.5, the command will suggest the peak only if a spectrum for *m/z* 1677.9 → 1384.5 has not yet been input via the **AddSpectrumFile** command.

Figure 40 summarizes the processing done by IDA's SuggestPeaks command.

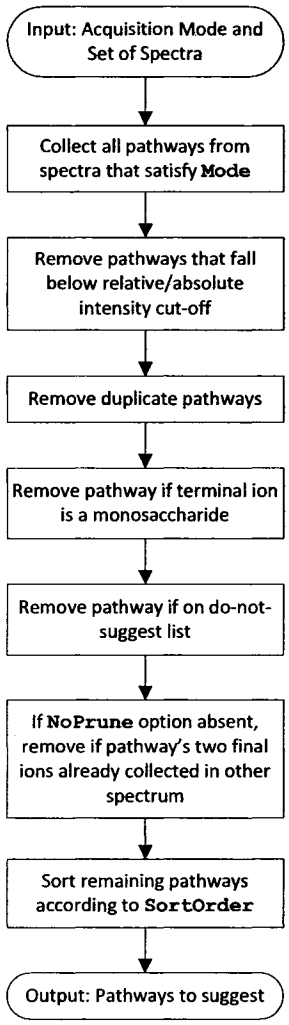


Figure 40: An overview of the Intelligent Data Acquisition algorithm as implemented by the SuggestPeaks command.

Typically, for *de novo* analysis, the analyst will collect the spectra suggested by the `SuggestPeaks Auto` command, add the spectra via `AddSpectrumFile`, and then repeat until `SuggestPeaks Auto` returns no further peaks. If the analyst is interested in collecting peaks that appear to come from unreported structural isomers, the `SuggestPeaks IsoDetect` command can be used. In fact, the vast majority of data collected for this work was collected using only the `Auto` and `IsoDetect` modes. The human has largely been removed from the data acquisition task.

The "scar" modes of Table 33 (namely, **`OnlyReducingEndScarred`**, **`OnlyNonReducingEndScarred`**, **`SingleSideScarred`**, and **`EneScar`**) are largely self-explanatory, returning peaks whose *m/z* values map to compositions which match the given scarring pattern. The other modes are described in detail below. For succinctness, we describe below only one scarring mode (**`OnlyNonReducingEndScarred`**).

8.2.1. The MajorGlyco Mode

All IDA modes examine the current set of spectra and suggest one or more ions to fragment. For the **`MajorGlyco`** mode, each spectrum is examined for the highest intensity peak whose putative composition can be explained as resulting from glycosidic cleavages; cross-ring cleavages and unknown compositions are ignored by this mode.

Because this mode returns the highest intensity peak at each step, in essence following the highest signal-to-noise ratio through the data, it generates the deepest possible MS^n pathway. This major glycosidic pathway becomes the backbone of the growing MS^n spectrum tree. Figure 41 shows the spectra suggest by five repeated applications of **`SuggestPeaks MajorGlyco`** to ion *m/z* 1851.9 as isolated from IgG. The first pathway suggested is 1851.9 → 1384.5, the second is 1851.9 → 1384.5 → 1125.4, and so on. The terminal spectrum in this tree, *m/z* 458.2, does not generate a suggested peak because the spectrum represents a disaccharide, and fragmenting the monosaccharide residues on that spectrum would not be structurally informative to GlySpy.

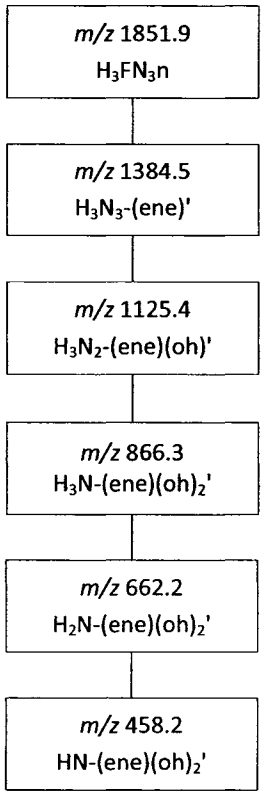


Figure 41: The MS^n spectrum tree built by repeated applications of the `SuggestPeaks MajorGlyco` command.

The spectra generated by this sequence of commands are available in Section A.3 as Spectrum A-28 through Spectrum A-33. Although the peaks returned at each stage happen to be the highest intensity peak on each spectrum, this is not always the case, especially with spectra whose precursor ion contains only a few residues.

8.2.2. The OnlyNonReducingEndScarred Mode

In the `OnlyNonReducingEndScarred` mode, as with all of the scarred modes, each spectrum is examined for the highest intensity peak that matches the given scarring pattern. As the name suggests, `OnlyNonReducingEndScarred` returns peaks whose compositions can be inferred to have scars only on their non-reducing end. When applied to the spectra shown in Figure 41, only a single pathway is returned: *m/z* 1851.9 → 1592.7. The result is shown in Spectrum A-34 and Figure 42. The terminal *m/z* 1592.7 is interpreted as having composition

H_3FN_2n-(oh). Because the reducing end residue n is included in the composition, IDA infers that the (oh) scar must be at the non-reducing end. All other peaks on the existing spectra have scars on both their reducing and non-reducing ends, or are below an intensity threshold, and so are not suggested in this mode.

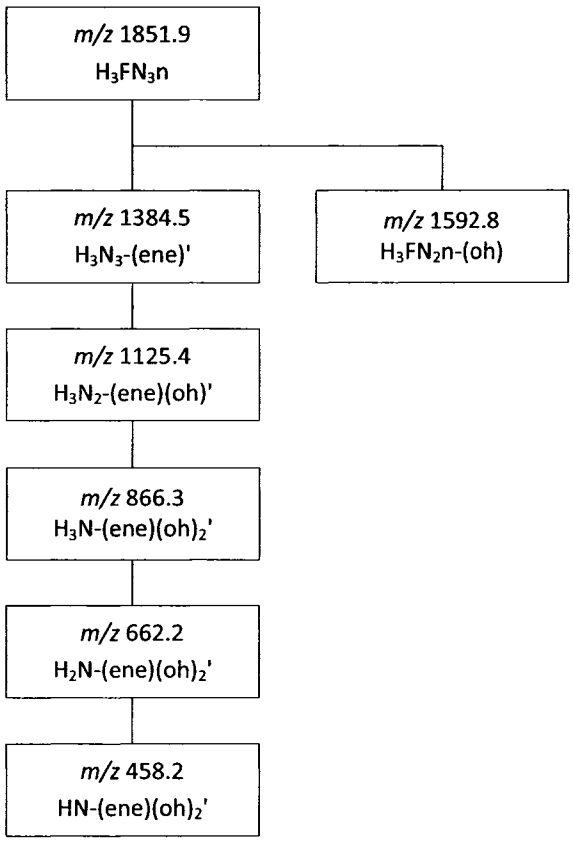


Figure 42: The MS^n spectrum tree after executing the `SuggestPeaks OnlyNonReducingEndScarred` command.

As is customary in this document, the major glycosidic pathway is shown as a vertical series of spectra. Spectra that branch off this backbone were suggested by other modes, such as, in this case, `OnlyNonReducingEndScarred`.

8.2.3. Pruning

If the **SuggestPeaks MajorGlyco** command were given again at this point, the result would be as shown in Figure 43 / Spectrum A-35. The ion *m/z* 1125.4 has been selected from the spectrum 1851.9 → 1592.8.

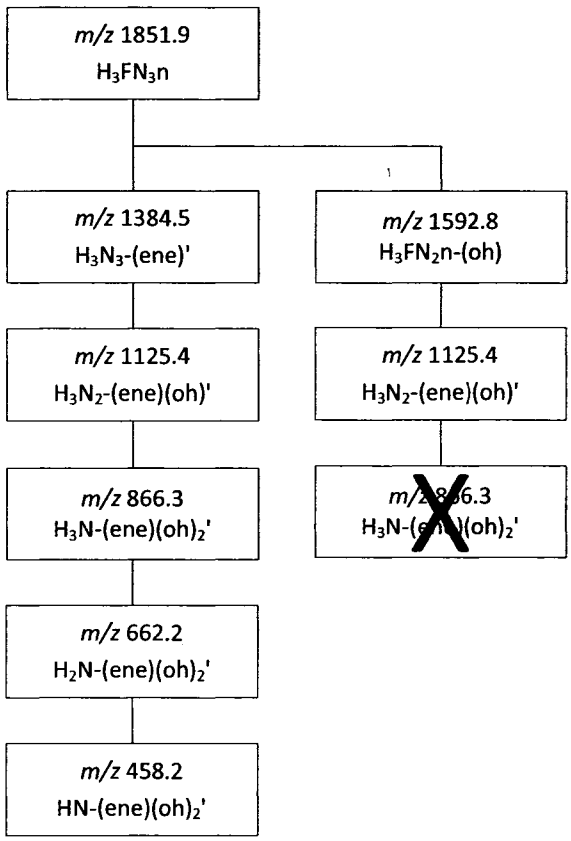


Figure 43: SuggestPeaks MajorGlyco adds the pathway *m/z* 1851.9 → 1592.8 → 1125.4, but pruning prevents the addition of the redundant *m/z* 866.3 spectrum.

One might expect that the next application of the **SuggestPeaks MajorGlyco** command would continue to extend the MS^n tree under the rightmost *m/z* 1125.4 spectrum. However, the spectra so collected would typically be of little value. They would merely mimic the *m/z* 866.3, 662.2, and 458.2 spectra already collected under the left branch. This occurs because the *m/z* 1125.4 ion in both branches specifies the same substructure. In the left branch we have lost a reducing end Fn disaccharide and then a terminal N; in the right branch the order of the

losses is reversed, but the result is the same. (Notice the extreme similarity between Spectrum A-30 and Spectrum A-35.)

To prevent the collection of these redundant spectra, IDA implements spectrum pruning. Suppose the **SuggestPeaks** command would return a pathway X → Y → Z, but a spectrum for W → Y → Z had already been collected. SuggestPeaks will notice that the two trailing ions Y → Z have been collected before from the precursor W, and will not suggest them again under the precursor X. Note, though, that the spectrum for Y is collected twice (as X → Y and W → Y), allowing for manual inspection to confirm their equivalence.

Spectrum pruning is an important technique for keeping the MS^n spectrum trees of manageable size. Without pruning, every branch in the tree would lead to the collection of many redundant spectra. However, in cases where the analyst decides that these spectra should be collected, the **NoPrune** option may be given to the **SuggestPeaks** command.

8.2.4. The MissingComplements Mode

Ions sometimes fragment into complementary pairs of products. For example, if the masses of two product ions sum to the mass of the precursor, a natural assumption is that the precursor has cleaved into two complementary fragments. More correctly, because both product ions contain a charge adduct, a complementary relationship is revealed when

Precursor – Charge Adduct = Product1 – Charge Adduct + Product2 – Charge Adduct

holds true within the user-specified mass error tolerance.

If one of the product ions has been fragmented, it is often informative to isolate and fragment the complementary ion, if possible. This is the function of the **MissingComplements** mode and is illustrated in Figure 44. Ion *m/z* 1592.8 lost an Fn disaccharide to form the *m/z* 1125.4 product. Because this example uses a sodium ion (mass: 23.0 Da) as the charge adduct,

SuggestPeaks determines that the complementary ion, if present on the *m/z* 1592.8 spectrum, would have an *m/z* of approximately 490.4. (Solve 1592.8 – 23.0 = 1125.4 – 23.0 + Product2 – 23.0 for Product2.) Spectrum A-34 does in fact contain such an ion, though of very low intensity, and it is returned by the **SuggestPeaks** command.

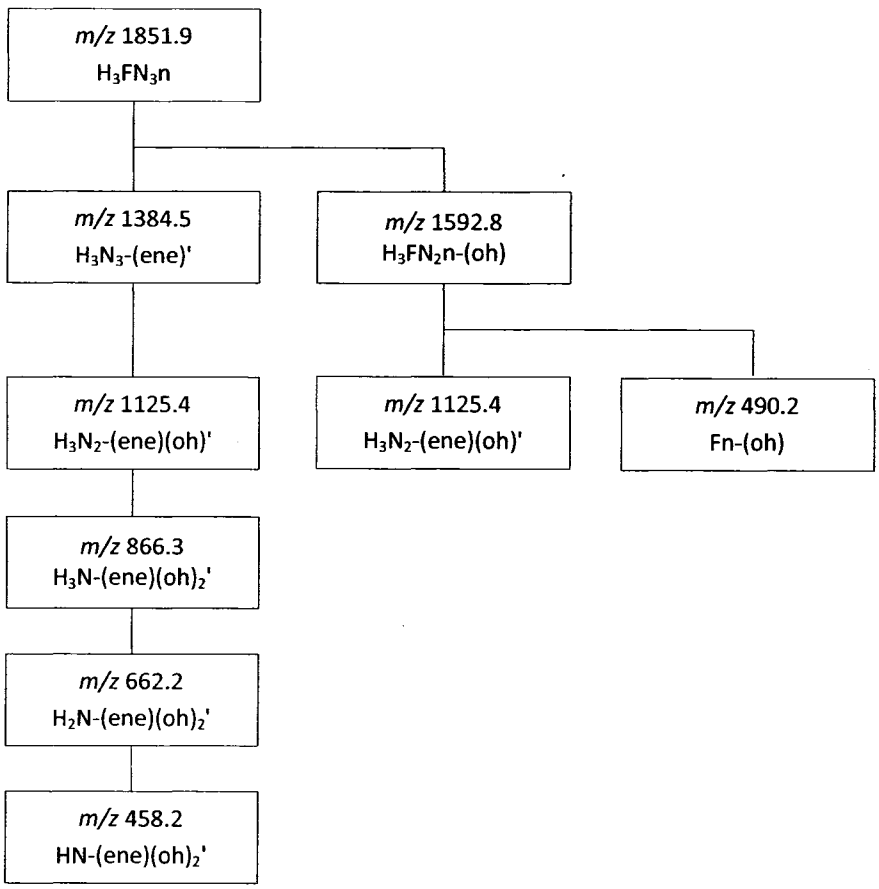


Figure 44: SuggestPeaks MissingComplements returns the pathway *m/z* 1851.9 → 1592.8 → 490.2 because *m/z* 490.2 and *m/z* 1125.4 appear to be complements of the precursor *m/z* 1592.8.

If the complementary nature of these products is not merely coincidental, then sequencing the *m/z* 1125.4 ion and the *m/z* 490.2 ion separately will reveal non-overlapping substructures within the glycan, which can then be combined to reveal the structure of the *m/z* 1592.8 ion. This is a good example of IDA ignoring many higher-intensity ions and ferreting out a low-intensity ion that promises to be structurally informative.

8.2.5. The Auto Mode

The preceding modes each have their strengths but share the weakness that the analyst must decide which mode to apply next. The **Auto** mode addresses this by integrating the most useful modes into one, moving GlySpy toward its goal of automated data collection.

Specifically, the **Auto** mode is a combination of the **MajorGlyco**, **OnlyReducingEndScarred**, **OnlyNonReducingEndScarred**, and **MissingComplements** modes. As shown in Figure 45, the **Auto** mode algorithm is as follows: If any **MajorGlyco** peaks have not been collected yet, suggest them. Otherwise, search for peaks that satisfy either **OnlyReducingEndScarred** or **OnlyNonReducingEndScarred**; if any are found, return them. Otherwise, search for a **MissingComplements** ion that is complementary to an existing spectrum; return it if found. Otherwise, return no pathways.

The pathways returned by **Auto** mode are subject to intensity cut-off limits, duplicate removal, and spectrum pruning.

In practice, high-quality MS^n spectrum trees can be collected merely by the mechanical repetition of the **SuggestPeaks Auto** command: apply the command, collect the spectra, add the spectra via **AddSpectrumFile**, and repeat until **SuggestPeaks Auto** returns no pathways. For example, applying this methodology to IgG glycan *m/z* 1851.9 will collect exactly the spectra shown in Figure 44. Significantly, these spectra are selected with no guidance from the analyst. Although the analyst is still required to physically acquire the suggested spectra, IDA has clearly moved us toward automating the data acquisition process.

One final detail requires attention. What if the pathway suggested by IDA's **Auto** mode cannot be obtained? Perhaps the mass spectrometer's sensitivity limit has been reached, or the analyst is dealing with a large legacy data set that does not contain the suggested pathway. In

this instance, the analyst may issue a **DoNotSuggestPathway** command. For example, **DoNotSuggestPathway 1851.9_1592.8_1125.4** would exclude that pathway (and all similar pathways within the user's selected mass error tolerance) from being suggested. This exclusion list applies to all of IDA's acquisition modes, but is most useful in preventing the **Auto** mode from becoming stuck in an infinite loop of asking for an unacquirable spectrum.

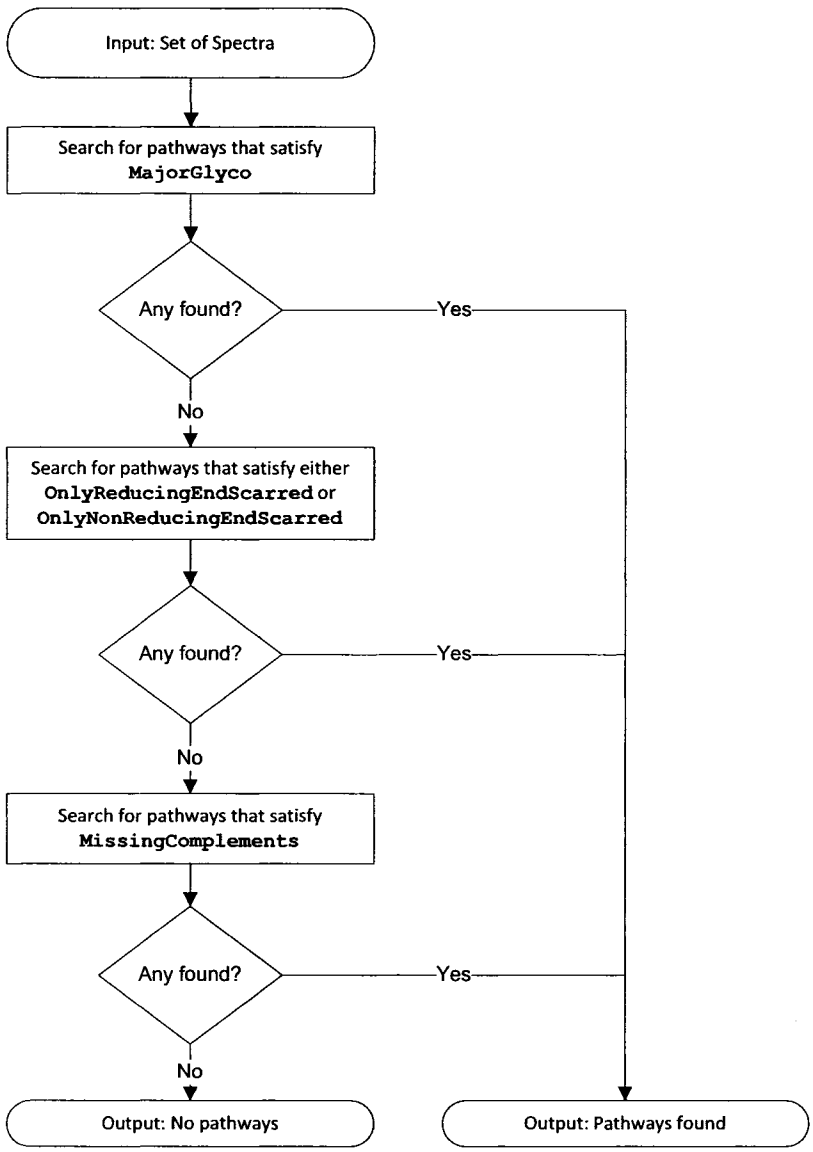


Figure 45: The SuggestPeaks Auto mode algorithm.

8.2.6. The IsoDetect Mode

Because GlySpy is a single application, it provides for rich communication between its component algorithms. In this case, IDA and IsoDetect combine to guide the analyst to acquire spectra for fragments that appear to come from structural isomers.

After collecting spectra via the **SuggestPeaks Auto** mode, the analyst might perform manual analysis with OSCAR or automated analysis with IsoSolve to identify structures that are likely to be present. Next, these structures and spectra are presented to IsoDetect, which returns a list of ions that appear to derive from structural isomers. **SuggestPeaks IsoDetect** will then apply intensity cut-offs, spectrum pruning, and so on, to generate a final list of isomeric pathways worthy of further examination. These pathways can then be selected for fragmentation via repeated application of the **SuggestPeaks Auto** command. Figure 46 illustrates this workflow.

A short example may clarify the procedure. In examining IgG ion *m/z* 1636.84, first the analyst applies **SuggestPeaks Auto** repeatedly to gather the initial set of spectra. IsoSolve returns NHH'(H')H'N'n' as a strong candidate structure. The analyst then performs these three commands to identify pathways that are incompatible with the candidate:

```
AddProposedGlycan NHH'(H')H'N'n'
IsoDetect NoCrossRing 1636.84 2 nul
SuggestPeaks IsoDetect 2 500 RankByEstTopologies 1636.84
```

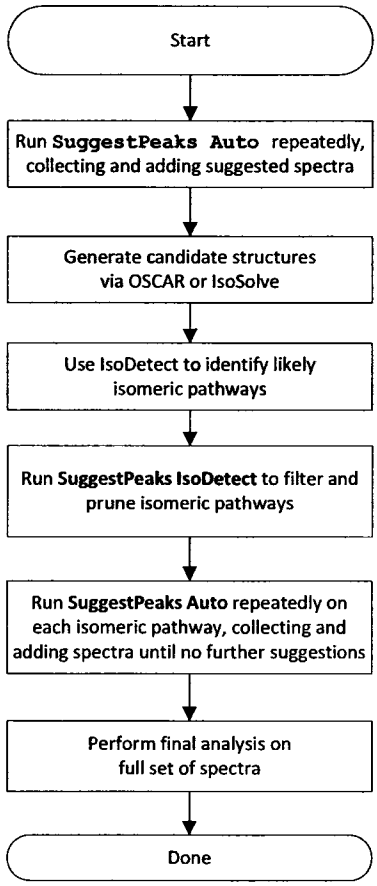


Figure 46: Integrating IsoDetect and SuggestPeaks to collect MS^n spectra for an isomeric mixture.

The final command returns the single pathway 1636.84_1139.45 as a likely isomeric fragment. (The ion *m/z* 1139.45 is identified to have the composition H_3N_2-(ene)', which is incompatible with the candidate.) This pathway then becomes a parameter to successive invocations of **`SuggestPeaks Auto`**:

```
; Ask for suggestions anywhere under the 1636.84_1139.45 spectrum
SuggestPeaks Auto 2 500 RankByEstTopologies 1636.84_1139.45

; Returned 1636.84_1139.45_880.32, so collect and add that spectrum
AddSpectrumFile IGG_1636_1139_880.raw

; Run SuggestPeaks on 1636.84_1139.45 again
SuggestPeaks Auto 2 500 RankByEstTopologies 1636.84_1139.45

; Returned 1636.84_1139.45_880.32_676.34, so add that spectrum, too
AddSpectrumFile IGG_1636_1139_880_676.raw

; Run SuggestPeaks on 1636.84_1139.45 again
SuggestPeaks Auto 2 500 RankByRelInt 1636.84_1139.45

; No peaks suggested, so done adding spectra
```

Note that all of the SuggestPeaks commands in this example use 1636.84_1139.45 as the pathway parameter. This restricts suggested pathways to those with that prefix, which in this case means we are collecting pathways to elucidate a structure that is incompatible with the candidate NHH'(H')H'N'n'. SuggestPeaks has harnessed the output of IsoDetect to guide its suggestions toward these isomeric structures.

8.3. Validation

The specific spectra collected by IDA for a variety of glycan samples are detailed in Chapter 9: AUTOMATED GLYCAN TOPOLOGY ANALYSIS. There we list the spectra collected for each sample and discuss why each spectrum was selected. Suggestions for improving IDA are also given.

CHAPTER 9:

AUTOMATED GLYCAN TOPOLOGY ANALYSIS

9.1. Overview

In this chapter we present analytical results for a series of glycans isolated from a variety of biological sources. In each section below, we use Intelligent Data Acquisition to collect a set of spectra for a given *m/z*. These spectra are then analyzed by IsoSolve, and the proposed structures are listed. These structures are then compared to the expected structures at that *m/z*. The list of expected structures from IgG is taken from Table II of (Butler 13); expected ovalbumin structures are from (Harvey 37); GM1a and GM1b are from (Svennerholm 80). These previously-reported structures are collected in Table 4 on page 27. For each example, we discuss how well GlySpy performed, indicate difficulties encountered and suggest future improvements.

It bears repeating that the vast majority of the analysis in this chapter was performed without human intervention. Exceptions to this will be noted.

9.2. Results and Discussion

For each in a series of glycans, we applied Intelligent Data Acquisition to collect a series of spectra, and then submitted these spectra to IsoSolve for structural analysis. The resulting topologies are evaluated for correctness, with interesting results discussed in detail.

Table 35 summarizes the main Intelligent Data Acquisition results, and Table 36 does the same for IsoSolve. Notice should be taken of the execution times in both tables, as they indicate that GlySpy's performance may enable it to be the centerpiece of a high-throughput glycomics analysis platform.

Source	*m/z*	Expected Structure(s)	Document Section	Number of Spectra Collected	Execution Time (seconds)
Bovine Brain Gangliosides	1273.65	GM1a: HN(S)HH-(oh) GM1b: SHNHH-(oh)	9.2.1.1	8	1.06
IgG	1606.83	NH'(H')H'N'(F)n'	9.2.2.1	11	0.25
	1636.84	HNH'(H')H'N'n'	9.2.3.1	7	0.11
	1677.87	NH'(NH')H'N'n'	9.2.4.1	12	0.74
	1810.93	HNH'(H')H'N'(F)n	9.2.5.1	8	0.25
	1851.96	NH'(NH')H'N'(F)n'	9.2.6.1	9	0.54
Ovalbumin	1187.61	H'(H')H'N'n'	9.2.7.1	9	0.11
	1636.84	NH'(HH')H'N'n'	9.2.8.1	11	0.71
	1677.87[1]	NH'(N)(H')H'N'n'	9.2.9.1	17	3.28
	1677.87	NH'(N)(H')H'N'n'	9.2.9.1	12	2.86
	1922.99	N(N)H'(N)(H')H'N'n' NH'(NH')(N)H'N'n'	9.2.10.1	14	3.95

Table 35: Summary of IDA results and execution times.
[1] The NoPrune option was specified for this test.

Source	*m/z*	Expected Structures	Doc. Section	Number of Pathways	Number of Structures Proposed	Ranking of Expected Structures	Execution Time (secs)
BBG	1273.65	HN(S)HH-(oh) SHNHH-(oh)	9.2.1.2	32	11	2 and 3	1.89
IgG	1606.83[1]	NH'(H')H'N'(F)n'	9.2.2.2	68	27	1	10.91
	1606.83	NH'(H')H'N'(F)n'	9.2.2.2	60	4	1	1.22
	1636.84	HNH'(H')H'N'n'	9.2.3.2	23	2	N/A	0.55
	1677.87	NH'(NH')H'N'n'	9.2.4.2	47	2	1	0.94
	1810.93	HNH'(H')H'N'(F)n	9.2.5.2	28	11	N/A	1.72
	1851.96	NH'(NH')H'N'(F)n'	9.2.6.2	41	1	1	0.46
OVA	1187.61[1]	H'(H')H'N'n'	9.2.7.2	49	10	1	1.80
	1636.84	NH'(HH')H'N'n'	9.2.8.2	67	12	6	4.54
	1677.87	NH'(N)(H')H'N'n'	9.2.9.2	61	5	2	2.98
	1922.99	N(N)H'(N)(H')H'N'n' NH'(NH')(N)H'N'n'	9.2.10.2	74	5	1 and 5	6.46

Table 36: Summary of IsoSolve results and execution times.
BBG = Bovine Brain Gangliosides, OVA = Ovalbumin.
[1] Test of *N*-linked structures executed without the –NLinkedBranching switch.

9.2.1. GM1a/GM1b *m/z* 1273.65

9.2.1.1 IDA

Listing 20 shows a typical input script used to collect spectra via GlySpy's Intelligent Data Acquisition module. The listing begins by giving the `-ReducingEndResidue unreduced` option, which specifies that only unreduced residues are to be considered as the reducing-end sugar. Next, `-UnmethylatedReducingEnd` specifies that the reducing end carbon of this structure is not methylated. In this case, it is a hydroxyl group, a scar left behind by the cleavage of this glycan from the ganglioside. See Figure 11.

Next, **AddSpectrumFile GM1ab_1877_1273.raw** is given, adding the spectrum generated from the fragmentation of the *m/z* 1273.65 ion. This spectrum will be the starting point for future spectrum suggestions. The script then issues a **SuggestPeaks Auto 2 100 RankByRelInt 1273.65** command, which returns the output shown in Listing 21. The parameters **2** and **100** specify cut-offs for relative and absolute intensity, respectively, and **RankByRelInt** sorts the suggested peaks by relative intensity. IDA suggests that the analyst acquire a spectrum for *m/z* 1273.40_898.21 and, as an aid for the analyst, lists the compositions of these ions as well. At this point the analyst acquires the spectrum and saves it to the file GM1ab_1877_1273_898.raw, to be added by the next **AddSpectrumFile** command.

This cycle–**SuggestPeaks Auto**, collect spectrum, **AddSpectrumFile**–is repeated until **SuggestPeaks** returns no suggested peaks. A summary of all *m/z* 1273 spectra collected using this method is given in Table 37 and the spectra themselves are presented as Spectrum A-10 through Spectrum A-17. The spectra are listed in the order in which they were suggested.

```
-ReducingEndResidue unreduced
-UnmethylatedReducingEnd

; Add the MS2 spectrum to start
AddSpectrumFile GM1ab_1877_1273.raw

; Run "SuggestPeaks Auto" to select next spectra to collect.
SuggestPeaks Auto 2 100 RankByRelInt 1273.65

; Add the suggested spectra and repeat SuggestPeaks
; until no further suggestions
AddSpectrumFile GM1ab_1877_1273_898.raw
SuggestPeaks Auto 2 100 RankByRelInt 1273.65

AddSpectrumFile GM1ab_1877_1273_898_486.raw
SuggestPeaks Auto 2 100 RankByRelInt 1273.65

AddSpectrumFile GM1ab_1877_1273_847.raw
AddSpectrumFile GM1ab_1877_1273_898_435.raw
SuggestPeaks Auto 2 100 RankByRelInt 1273.65

AddSpectrumFile GM1ab_1877_1273_847_472.raw
SuggestPeaks Auto 2 100 RankByRelInt 1273.65

; SuggestPeaks wants 1273.40_449.14, but since this is a
; legacy data set and that spectrum was not collected,
; we substitute a close match instead: 1273.5_898.3_449.2
DoNotSuggestPathway 1273.40_449.14
AddSpectrumFile GM1ab_1877_1273_898_449.raw
SuggestPeaks Auto 2 100 RankByRelInt 1273.65

AddSpectrumFile GM1ab_1877_1273_898_472.raw
SuggestPeaks Auto 2 100 RankByRelInt 1273.65

; No further peaks suggested
```

Listing 20: The input script used to collect the spectra of Table 37.
IDA's SuggestPeaks command is used to repeatedly request new spectra collect.
The process concludes when SuggestPeaks Auto returns no spectra.

```
******* Begin SuggestPeaks Results (1 Peak Found) *******
--------------------------------------------------
Peak 1 of 1:
iASP: 0   RelInten: 100.0000   AbsInten: 3.44E+04   good   EstTop: <nil>
Spectrum: 0
Pathway: 1273.40_898.21[100.0000%]
Spectrum name: GM1ab_1877_1273.raw
Spectrum scan: 1

PathwayComps:
#Ions: 2

Ion 0 has 1 possible composition:
MSn: 1273.62 H3NS-(oh)'

Ion 1 has 1 possible composition:
MSn:  898.43 H3N-(oh)2'
--------------------------------------------------
******* End SuggestPeaks Results *******
```

Listing 21: Sample output for the first SuggestPeaks command in Listing 20.

Table 37: Spectra suggested by IDA for GM1a/GM1b *m/z* 1273.65.

#	Spectrum Added	Relative Intensity	Putative Terminal Ion Compositions	Reason Added
1	**1274.40**	N/A	1273.62 H_3NS-(oh)'	Initial
2	1273.40_**898.21**	100.0	898.43 H_3N-$(oh)_2$'	Major(1)
3	1273.40_898.30_**486.07**	100.0	486.23 HN-(ene)'	Major(2)
4	1273.40_**847.20**	29.5	847.41 HNS-(ene)'	REScar(1)
5	1273.50_898.30_**435.12**	23.0	435.18 H_2-$(oh)_3$'	Complement(3)
6	1273.40_847.30_**472.05**	100.0	472.22 HN-(ene)(oh)'	Major(4)
7	1273.5_898.3_**449.2**	2.9	449.20 H_2-$(oh)_2$'	Complement(4)
8	1273.50_898.30_**472.05**	23.2	472.22 HN-(ene)(oh)'	Complement(7)

IDA had requested 1273.40_449.14 for spectrum number 7 of Table 37, serving as a complement to spectrum 4. However, this is a legacy data set and the samples were unavailable to collect this spectrum. However, a close match was available, and so 1273.5_898.3_449.2 was

used instead. A `DoNotSuggestPathway 1273.40_449.14` command was then given to prevent IDA from continuously requesting this unacquirable spectrum.

The "Reason Added" column of Table 37 explains why IDA selected the spectrum for inclusion. The selection criteria are detailed in Table 38.

Table 38: Shorthand for IDA's spectrum selection criteria.

Abbreviation	Explanation
Initial	The initial fragmentation spectrum of the target structure.
Major(*n*)	The highest-intensity glycosidic peak on spectrum *n*.
Complement(*n*)	The complement of spectrum *n*'s terminal ion.
REScar(*n*)	A fragment from spectrum *n* that has only reducing-end scars.
NREScar(*n*)	A fragment from spectrum *n* that only has non-reducing-end scars.

From this we see that IDA successfully identified several complementary ions (entries 5, 7, and 8 in Table 37, complementary to entries 3, 4, and 7, respectively). For example, entries 3 (1273.40_898.30_486.07) and 5 (1273.50_898.30_435.12) are complementary because the terminal ions (486.07 and 435.12) sum to their shared precursor (898.30), after the sodium adducts (22.99 Da on each ion) are accounted for. Complementary ions can be extremely valuable when making structural assignments, but are difficult or impossible to collect using other automated data acquisition tools.

Even experienced analysts may miss low intensity complementary peaks. The requested peak `1273.40_449.14` has a relative intensity of only 2.9%, so low that it is not even labeled on Spectrum A-10—and, more to the point, so low that the analyst who collected this legacy data set did not fragment it, despite collecting literally dozens of spectra. Clearly, automated identification of complementary fragment ions is a valuable capability.

We also see that entry 4 was selected because it contained only a reducing-end scar. As such, IDA could surmise that this fragment represented a terminal branch, which is another desirable ion for structural assignment. This pathway was extended by entry 6, chosen because it was the major glycosidic peak on entry 4's spectrum.

9.2.1.2 IsoSolve

IsoSolve was executed using the input shown in Listing 22. First, the appropriate options are set: `-ReducingEndResidue unreduced` and `-UnmethylatedReducingEnd`. Next, the spectrum files collected by IDA in the previous section are each added via `AddSpectrumFile`. (The `DoNotSuggestPathway` command is included for symmetry with the IDA process, but does not affect the results of this test.) Lastly, `IsoSolve NoCrossRing 1273.65 2` is executed. Here, the parameter `2` means to consider all pathways at or above a 2% relative intensity cut-off. Given that input, IsoSolve returns 11 structures from 32 total pathways. See Table 39.

```
-ReducingEndResidue unreduced
-UnmethylatedReducingEnd

; Begin by adding all of the spectra returned by "SuggestPeaks Auto"
AddSpectrumFile GM1ab_1877_1273.raw
AddSpectrumFile GM1ab_1877_1273_898.raw
AddSpectrumFile GM1ab_1877_1273_898_486.raw
AddSpectrumFile GM1ab_1877_1273_847.raw
AddSpectrumFile GM1ab_1877_1273_898_435.raw
AddSpectrumFile GM1ab_1877_1273_847_472.raw
DoNotSuggestPathway 1273.40_449.14
AddSpectrumFile GM1ab_1877_1273_898_449.raw
AddSpectrumFile GM1ab_1877_1273_898_472.raw

; Run IsoSolve
IsoSolve NoCrossRing 1273.65 2
```

Listing 22: Input to execute IsoSolve on the spectra collected by IDA.

Table 39: IsoSolve results for the spectra shown in Table 37.

#	Score	Linear Code	Compatible Pathways	Cumulative
1	83.33	HNH(S)H-(oh)	20/24	62.50% (20/32)
2*	75.00	HN(S)HH-(oh)	15/20	65.63% (21/32)
3*	73.08	SHNHH-(oh)	19/26	100.00% (32/32)
4	71.43	SNH(H)H-(oh)	20/28	100.00% (32/32)
5	69.23	SHN(H)H-(oh)	18/26	100.00% (32/32)
6	60.71	S(N)HHH-(oh)	17/28	100.00% (32/32)
7	59.09	NHH(S)H-(oh)	13/22	100.00% (32/32)
8	46.15	S(H)NHH-(oh)	12/26	100.00% (32/32)
9	45.45	HNSHH-(oh)	5/11	100.00% (32/32)
10	45.45	NHS(H)H-(oh)	5/11	100.00% (32/32)
11	45.45	NHSHH-(oh)	5/11	100.00% (32/32)

The score associated with each structure represents the percentage of possibly compatible pathways that were actually compatible. (If a structure were compatible with 10 spectra containing a total of 100 pathways, but the structure was only compatible with 50 of these pathways, its score would be 50/100, or 50%.) This fraction is explicitly given in the Compatible Pathways column.

The Cumulative column indicates the percentage of all pathways compatible with this structure or any previous structure. In this case there are 32 total pathways, and the first structure is compatible with 20. Combined, structures 1 and 2 are compatible with 21 of the 32 pathways. Structures 1, 2, and 3 combined are compatible with all 32 pathways. This column shows how many of the proposed structures are required to cover any fraction of the available pathways. It could be used in the future as a cut-off for structure reporting, e.g., limiting

IsoSolve's output to the highest-scoring structures that collectively cover a user-selected percentage of the pathways.

9.2.1.3 Discussion

Structures 2 and 3, asterisked, represent the expected structures GM1a and GM1b, respectively. As such, IsoSolve combined with this simple scoring system were sufficient to place the correct structures in two of the top three output slots.

The scoring system could be improved by looking for fragments arising from expected facile cleavages. For example, structure 4, SNH(H)H-(oh), would be expected to cleave at the N, yielding SN-(ene) and H_3-$(oh)_2$ fragments, leading to the observed fragments *m/z* 643.31 and *m/z*, 653.29 respectively. Since neither of these fragments is present in Spectrum A-10, structure 4 should be suitably penalized. Further, this improved scoring system could be implemented for every spectrum gathered. That is, if a spectrum was consistent with a given structure, then it should be examined for expected fragments, and the more that are missing, the higher the penalty.

Interestingly, this extended scoring strategy would *not* penalize structure 1, HNH(S)H-(oh). From the facile HexNAc cleavage, we would expect to find fragments HN-(ene) and H_2S-$(oh)_2$ as ions *m/z* 486.23 and *m/z* 810.37, respectively. However, we do find those fragments, both from structure 2! Here, only the location of the S has changed, but this is not revealed by the single HexNAc cleavage. Fragmenting the H_2S-$(oh)_2$ ion would also fail to distinguish between structures 1 and 2, as they would yield the same expected fragments: H_2-$(oh)_2$, S-(ene), and H-$(oh)_2$. This analysis is greatly complicated by the fact that the reducing-end H is not reduced, and comes with a reducing-end scar. If the reducing-end scar were not present, for example,

the fragment HS-(oh) would be diagnostic of structure 1, while H_2-(oh) would be diagnostic of structure 2.

However, despite these limitations, IsoSolve identified the GM1a and GM1b isomers as two of the top three suggested structures. Eliminating structure 1 from contention using mass spectrometric techniques only would be beyond the capability of many human analysts.

9.2.2. IgG *m/z* 1606.83

9.2.2.1 IDA

Intelligent Data Acquisition was performed on IgG ion *m/z* 1606.83. This and all subsequent IDA examples followed the same data acquisition process as described in Section 9.2.1.1: Run **`SuggestPeaks Auto`**, manually collect the spectrum, add via **`AddSpectrumFile`**, and repeat until **`SuggestPeaks`** runs dry.

Examining Table 40, we see that the initial spectrum spawned a deep MSn probe into the structure, yielding entries 2-5 as MajorGlyco fragments. Interestingly, entries 8 and 9 were both selected as reducing-end-scar (REScar) fragments from *m/z* 1139.5. Typically, the only the highest intensity REScar fragment is reported, but in this case, there were two spectrum scans available. On one scan *m/z* 486.17 was slightly more abundant than *m/z* 912.35, but on the other scan, the relative intensities were reversed. As seen in Table 40, these fragments' intensities are extremely similar.

The initial ion *m/z* 1606.83 represents a glycan with seven residues. From this a total of 11 spectra were collected, a reasonable number for a structure of this size. A single spectrum typically requires 15 to 30 seconds of acquisition time, so the entire spectrum set represents approximately three to five minutes of instrument usage. However, because the analyst must

manually transcribe suggested *m/z* pathways to acquire spectra, the actual data acquisition time was much longer.

Table 40: Spectra suggested by IDA for IgG *m/z* 1606.83.

#	Spectrum Added	Relative Intensity	Putative Terminal Ion Compositions	Reason Added
1	**1606.83**	N/A	1606.83 H_2FN_3h 1606.83 H_3FN_2n	Initial
2	1606.83_**1139.46**	100.0	1139.56 H_3N_2-(ene)'	Major(1)
3	1606.83_1139.55_**880.34**	100.0	880.42 H_3N-(ene)(oh)'	Major(2)
4	1606.83_1139.55_880.50_ **676.26**	100.0	676.32 H_2N-(ene)(oh)'	Major(3)
5	1606.83_1139.55_880.50_676.36_ **431.12**	100.0	431.19 H_2-(ene)(oh)'	Major(4)
6	1606.83_**1347.54**	38.5	1347.69 H_2FN_2h-(oh) 1347.69 H_3FNn-(oh)	REScar(1)
7	1606.83_**490.20**	3.7	490.23 HN-$(oh)_2$' 490.26 Fn-(oh)	Complement(2)
8	1606.83_1139.55_**486.17**	2.6	486.23 HN-(ene)'	REScar(2)
9	1606.83_1139.55_**912.35**	2.6	912.44 H_3N-(oh)'	REScar(2)
10	1606.83_1139.46_912.35_ **653.27**	100.0	653.30 H_3-$(oh)_2$'	Major(9)
11	1606.83_1347.60_**880.32**	100.0	880.42 H_3N-(ene)(oh)' 880.45 HFNh-(ene) 880.45 H_2Fn-(ene)	Major(6)

9.2.2.2 IsoSolve

For the collected spectra in Table 40, we executed two separate IsoSolve tests. The results summarized in Table 41 were generated without using the –NLinkedBranching switch, and hence produced structures that do not contain the H_3Nn core. By contrast, the results of Table

42 were generated with the use of the –NLinkedBranching switch. (Analyses of all *N*-linked structures in this document use the switch unless otherwise stated.)

Table 41: IsoSolve results for the spectra shown in Table 40. The –NLinkedBranching switch was *not* used for this analysis.

#	Score	Linear Code	Compatible Pathways	Cumulative
1*	80.88	NH(H)HN(F)n	55/ 68	80.88% (55/68)
2	78.69	NN(H)HH(F)n	48/ 61	86.76% (59/68)
3	69.49	HN(NH)H(F)n	41/ 59	89.71% (61/68)
4	67.80	NH(N)(H)H(F)n	40/ 59	92.65% (63/68)
5	64.00	HN(H)HN(F)n	16/ 25	92.65% (63/68)
6	64.00	HNHHN(F)n	16/ 25	92.65% (63/68)
7	63.49	NHN(H)HFn	40/ 63	94.12% (64/68)
8	58.82	NNH(H)HFn	30/ 51	94.12% (64/68)
9	57.35	NHHHN(F)n	39/ 68	95.59% (65/68)
10	56.76	NHNH(NF)h	21/ 37	97.06% (66/68)
11	52.63	N(H)NHHFn	30/ 57	98.53% (67/68)
12	51.35	N(N)H(F)(H)Hn	19/ 37	98.53% (67/68)
13	51.35	NHNN(F)(H)h	19/ 37	98.53% (67/68)
14	51.35	NHN(NF)(H)h	19/ 37	98.53% (67/68)
15	51.35	NHNF(N)(H)h	19/ 37	98.53% (67/68)
16	48.65	NHNN(F)Hh	18/ 37	98.53% (67/68)
17	48.65	NHN(NFH)h	18/ 37	98.53% (67/68)
18	48.65	NHNFH(N)h	18/ 37	98.53% (67/68)
19	47.54	NNHHHFn	29/ 61	98.53% (67/68)
20	47.06	HHH(N)N(F)n	24/ 51	100.00% (68/68)
21	46.03	NHNHHFn	29/ 63	100.00% (68/68)
22	45.95	NHNNH(F)h	17/ 37	100.00% (68/68)
23	43.24	N(N)HN(F)(H)h	16/ 37	100.00% (68/68)
24	43.24	NHNH(FN)h	16/ 37	100.00% (68/68)
25	42.11	N(N)HHHFn	24/ 57	100.00% (68/68)
26	40.54	NNHNH(F)h	15/ 37	100.00% (68/68)
27	32.43	N(H)N(N)(F)Hh	12/ 37	100.00% (68/68)

Table 42: IsoSolve results for the spectra shown in Table 40. The –NLinkedBranching switch was used for this analysis.

#	Score	Linear Code	Compatible Pathways	Cumulative
1*	91.67	NH'(H')H'N'(F)n'	55/60	91.67% (55/60)
2	63.64	NH'(H')H'(F)N'n'	14/22	98.33% (59/60)
3	56.52	N(F)H'(H')H'N'n'	13/23	98.33% (59/60)
4	47.83	H'(H')H'(N)N'(F)n'	22/46	100.00% (60/60)

9.2.2.3 Discussion

A comparison of these results is instructive. Notably, the expected structure was ranked number one in both instances. However, in the first case, IsoSolve returned 27 structures from 68 total pathways; in the second, only four structures from 60 total pathways. Not only does the –NLinkedBranching switch lead to a marked reduction in the number of structures produced, it also reduces the total number of pathways considered. In this case, eight pathways were excluded by the switch, indicating the likely presence of structures that do not contain the expected core residues; these non-canonical structures are likely the cause for at least some of the additional entries in Table 41. We will discuss these unexpected structures in more detail later.

We can imagine applying the additional scoring heuristics to the entries of Table 42. For example, entries 2 and 3 would be expected to have a facile cleavage at the penultimate N, losing only the reducing-end n and yielding a fragment with composition H_3FN_2-(ene), *m/z* 1313.64. However, no such peak is apparent in Spectrum A-18 (page 217), effectively ruling out these structures.

Entries 1 and 4, by contrast, would lose both the terminal n and the attached F with this cleavage, leaving an expected fragment of H3N2-(ene), *m/z* 1139.56. This is the base peak of

the spectrum, giving greater weight to these structures. However, structure 1's score is so much higher than 4's that it would rightly be considered the best solution.

9.2.3. IgG *m/z* 1636.84

9.2.3.1 IDA

Table 43 summarized the spectra collected for IgG ion *m/z* 1636.84. We again see a MajorGlyco pathway extended (entries 1 through 5). Entry 6 collects a fragment with only a reducing-end scar, and entry 7 extends that spectrum with a single MajorGlyco spectrum. These spectra are presented as Spectrum A-19 through Spectrum A-25 beginning on page 218.

Table 43: Spectra suggested by IDA for IgG *m/z* 1636.84.

#	Spectrum Added	Relative Intensity	Putative Terminal Ion Compositions	Reason Added
1	**1636.84**	N/A	1636.84 H_3N_3h 1636.84 H_4N_2n	Initial
2	1636.84_**1173.49**	100.0	1173.60 H_2N_2h-(oh) 1173.60 H_3Nn-(oh)	Major(1)
3	1636.84_1173.65_**914.37**	100.0	914.46 H_2Nh-$(oh)_2$ 914.46 H_3n-$(oh)_2$	Major(2)
4	1636.84_1173.65_914.45_**710.30**	100.0	710.36 HNh-$(oh)_2$ 710.36 H_2n-$(oh)_2$	Major(3)
5	1636.84_1173.65_914.45_710.36_**506.20**	100.0	506.26 Nh-$(oh)_2$ 506.26 Hn-$(oh)_2$	Major(4)
6	1636.84_**1343.50**	14.3	1343.66 H_4N_2-(ene)'	REScar(1)
7	1636.84_1343.50_**880.40**	100.0	880.42 H_3N-(ene)(oh)'	Major(6)

9.2.3.2 IsoSolve

Using the spectra from Table 43, IsoSolve returns 2 structures from 23 total pathways (Table 44).

Table 44: IsoSolve results for the spectra shown in Table 43.

#	Score	Linear Code	Compatible Pathways	Cumulative
1	95.65	NHH'(H')H'N'n'	22/23	95.65% (22/23)
2	40.00	NH'(H')H'N'(H)n'	6/15	100.00% (23/23)

9.2.3.3 Discussion

In this instance, IsoSolve failed to produce the expected structure, HNH'(H')H'N'n'. This surprising result stems from an equally surprising fact: The major glycosidic disassembly pathway, *m/z* 1636.84 → 1173.65 → 914.45 → 710.36 → 506.20 (Table 43, entries 1 through 5), is not consistent with *any* glycan containing the *N*-linked core! A look at the possible compositions of these ions reveals why. First, we must disregard compositions containing h, a reduced hexose, if we expect to find an *N*-linked core. Now we see that the composition sequence, including the major peak of *m/z* 316.2 from the final spectrum, must be as shown in Table 45.

Table 45: Putative composition pathway for the non-*N*-linked structure found at IgG *m/z* 1636.84.

#	*m/z*	Composition
1	1636.84	H_4N_2n
2	1173.60	H_3Nn-(oh)
3	914.46	H_3n-$(oh)_2$
4	710.36	H_2n-$(oh)_2$
5	506.26	Hn-$(oh)_2$
6	316.17	n-(oh)

As one example, the H_3n-$(oh)_2$ composition for *m/z* 914.46 cannot possible come from an *N*-linked glycan because there is no intervening N between the H_3 and the reducing-end n. The incompatibility of this pathway with the expected *N*-linked core is confirmed by OSCAR. Executing the input script shown in Listing 23 produces no candidate structures.

```
-NLinkedBranching
AddPathway NoCrossRing 1636.84_1173.65_914.45_710.36_506.20_316.2
Summarize
```

Listing 23: Input that yields no candidate *N*-linked structures.

Interpreting this pathway as containing a reduced hexose (h) fails to yield a composition assignment at every ion: *m/z* 1636.84 = H_3N_3h, *m/z* 1173.60 = H_2N_2h-(oh), *m/z* 914.46 = H_2Nh-$(oh)_2$, and *m/z* 710.36 = HNh-$(oh)_2$, but *m/z* 506.26 and *m/z* 316.17 have no possible glycosidic compositions.

Clearly, something unexpected has been found here. This provides an opportunity to show how an analyst might use GlySpy to manually assign a novel structure.

Executing the input of Listing 24 produced a list of only five possible structures (Table 46).

```
-ReducingEndResidue n
AddPathway NoCrossRing 1636.84_1173.65_914.45_710.36_506.20_316.2
Summarize
```

Listing 24: Input that reveals only five possible structures for the pathway 1636.84_1173.65_914.45_710.36_506.20_316.2. None of these structures contain the expected *N*-linked core.

Table 46: The five structures generated by Listing 24.

#	Possible Structures
1	NHH(HN)Hn
2	HNH(NH)Hn
3	HNHH(N)Hn
4	NHHH(N)Hn
5	NHH(NH)Hn

At this point the analyst decides to collect more data for analysis. By manually adding several interesting spectra, the `SuggestPeaks Auto` command is then repeated until, in this case, eleven additional spectra have been acquired. This enhanced data set is then subjected to IsoSolve analysis both with (Table 47) and without (Table 48) the –NLinkedBranching option.

Table 47: IsoSolve results for the additional *m/z* 1636 spectra. The –NLinkedBranching switch was specified for this analysis.

#	Score	Linear Code	Compatible Pathways	Cumulative
1	93.55	NHH'(H')H'N'n'	29/31	80.56% (29/36)
2	87.10	HNH'(H')H'N'n'	27/31	83.33% (30/36)
3	54.55	NH'(H')H'N'(H)n'	12/22	100.00% (36/36)

Table 47 shows two high-scoring *N*-linked candidates. Entries 1 and 3 are repeated from entries 1 and 2 of Table 44, but we notice a new entry, HNH'(H')H'N'n', which matches the expected structure.

Table 48 shows the structures produced by IsoSolve without the –NLinkedBranching flag. The two highest-scoring topologies, HNH(NH)Hn and NHH(HN)Hn, are repeated from Table 46, lending them further credence. The 15 highest-scoring structures in Table 48 all share the same very unusual reducing-end motif of Hn. GlySpy has produced, with very little direction from the analyst, a short list of structures worthy of further investigation.

It should be noted that none of these structures would have been considered if GlySpy had implemented (presumed) biosynthetic constraints. This is both the strength and weakness of *de novo* analysis: structures are not incorrectly ruled out, but the generated list is necessarily longer.

Further complicating the analysis is the interaction of the structures with and without the expected *N*-linked core. For example, the expected *N*-glycan HNH'(H')H'N'n' would be expected to have a facile loss of the terminal HN and reducing-end n, yielding a composition of H_3N-(oh)(ene)', *m/z* 880.4. This ion is in fact present in Spectrum A-19, lending support to this structure. The highest-ranking *N*-glycan from Table 47, NHH'(H')H'N'n', would be expected to lose a terminal N and reducing-end n, giving a composition of H_4N-(ene)(oh)', *m/z* 1084.5. However, this ion is absent from Spectrum A-19, perhaps signifying that this structure is an artifact caused by combining pathways taken from two or more structural isomers.

Table 48: IsoSolve results for the additional *m/z* 1636 spectra. The –NLinkedBranching switch was *not* specified for this analysis.

#	Score	Linear Code	Compatible Pathways	Cumulative
1	87.88	HNH(NH)Hn	58/66	81.69% (58/71)
2	87.10	NHH(HN)Hn	54/62	83.10% (59/71)
3	81.97	HN(NH)HHn	50/61	90.14% (64/71)
4	80.65	HNH(N)(H)Hn	50/62	90.14% (64/71)
5	79.03	NHH(N)(H)Hn	49/62	91.55% (65/71)
6	77.78	NHH(NH)Hn	49/63	91.55% (65/71)
7	77.61	HNH(N)HHn	52/67	91.55% (65/71)
8	75.56	NHN(H)HHn	34/45	98.59% (70/71)
9	69.05	HNN(H)HHn	29/42	98.59% (70/71)
10	68.75	NHH(N)HHn	44/64	98.59% (70/71)
11	68.75	NNH(H)HHn	22/32	100.00% (71/71)
12	66.67	NH(H)HNHn	30/45	100.00% (71/71)
13	62.79	HNH(H)HNn	27/43	100.00% (71/71)
14	56.25	N(N)H(H)HHn	18/32	100.00% (71/71)
15	51.11	NHHHNHn	23/45	100.00% (71/71)
16	47.37	HN(N)HH(H)n	18/38	100.00% (71/71)
17	39.39	NHHNH(H)n	13/33	100.00% (71/71)

The analysis can be taken further, but several important points have been made. To wit:

1) IDA can be assisted by a human by providing "seed" pathways to be automatically extended.

2) Complicated isomeric mixtures are more common in biologically-derived samples than is generally appreciated and can greatly complicate structural analysis. In these cases,

chromatographic separation might be considered to reduce the complexity of the mixture.

3) Even under these adverse circumstances, GlySpy is able to quickly point the analyst toward novel structures, including those that would have been excluded by the application of biosynthetic constraints.

9.2.4. IgG *m/z* 1677.87

9.2.4.1 IDA

The spectra collected for IgG *m/z* 1677.87 are shown in Table 49. Again we see the typical MajorGlyco pathway extended in steps 1-6, with additional non-reducing-end scar fragments selected in steps 7, 9, and 11. Each of these in turn generates a single MajorGlyco spectrum (entries 8, 10 and 12, respectively) before pruning halts the data collection. A total of 12 spectra are collected for a structure containing seven residues.

Table 49: Spectra suggested by IDA for IgG *m/z* 1677.87.

#	Spectrum Added	Relative Intensity	Putative Terminal Ion Compositions	Reason Added
1	**1677.87**	N/A	1677.86 F_4S_2f 1677.87 H_2N_4h 1677.87 H_3N_3n	Initial
2	1677.87_**1384.53**	100.0	1384.68 H_3N_3-(ene)'	Major(1)
3	1677.87_1384.64_**1125.43**	100.0	1125.54 H_3N_2-(ene)(oh)'	Major(2)
4	1677.87_1384.64_1125.54_**866.33**	100.0	866.40 H_3N-(ene)$(oh)_2$'	Major(3)
5	1677.87_1384.64_1125.54_866.40_**662.22**	100.0	662.30 H_2N-(ene)$(oh)_2$'	Major(4)
6	1677.87_1384.64_1125.54_866.40_662.36_**458.13**	100.0	458.20 HN-(ene)$(oh)_2$'	Major(5)
7	1677.87_**1418.57**	96.3	1418.73 H_2N_3h-(oh) 1418.73 H_3N_2n-(oh)	NREScar(1)
8	1677.87_1418.60_**1125.38**	100.0	1125.54 H_3N_2-(ene)(oh)'	Major(7)
9	1677.87_1418.60_**1159.44**	54.7	1159.58 H_2N_2h-$(oh)_2$ 1159.58 H_3Nn-$(oh)_2$	NREScar(7)
10	1677.87_1418.60_1159.50_**866.31**	100.0	866.40 H_3N-(ene)$(oh)_2$'	Major(9)
11	1677.87_1418.60_1159.50_**900.43**	78.6	900.44 H_2Nh-$(oh)_3$ 900.44 H_3n-$(oh)_3$	NREScar(9)
12	1677.87_1418.60_1159.50_900.40_**696.36**	100.0	696.34 HNh-$(oh)_3$ 696.34 H_2n-$(oh)_3$	Major(11)

9.2.4.2 IsoSolve

Given those spectra, IsoSolve returns two structures from 47 total pathways. See Table 50. The expected structure, asterisked, is ranked as the higher-scoring of the two.

Table 50: IsoSolve results for the spectra shown in Table 49.

#	Score	Linear Code	Compatible Pathways	Cumulative
1*	97.87	NH'(NH')H'N'n'	46/47	97.87% (46/ 47)
2	91.30	NH'(N)(H')H'N'n'	42/46	100.00% (47/47)

9.2.4.3 Discussion

As Table 50 shows, the expected structure is compatible with 46 of 47 total pathways. The single available pathway with which it is not compatible is *m/z* 1677.87 → 1384.64 → 1125.54 → 866.40 → 662.36 → 444.11. The terminal *m/z* 444.11 ion has composition HN-(ene)$(oh)_3$' and is invariably found in structures which contain a bisecting HexNAc. (In structures without a bisecting HexNAc, the central core would yield ion *m/z* 458.20 with composition HN-(ene)$(oh)_2$'.)

Structure 2 is exactly the structure that would be expected if a bisecting HexNAc were present. A very strong case can be made that IsoSolve, with no human guidance whatsoever, has successfully identified a structural isomer not reported in (Butler 13). This feat is considerably more impressive when one considers Spectrum A-26. Here, *m/z* 444.1 has a relative intensity of only 2%, so small as to be unlabeled in the spectrum. Clearly the bisecting HexNAc isomer is of much lower abundance than the reported structure. The spectrum for the pathway 1677.87 → 1384.64 → 1125.54 → 866.40 → 662.36 → 444.11 was collected (Spectrum A-27) to prove the presumed bisecting profile. Although the spectrum has a very low normalization level (3.80E-2) and only one ion was measured (*m/z* 268.1), that ion was consistent with composition N-(ene)(oh)', which in turn is consistent with, but not diagnostic of, a bisecting HexNAc.

9.2.5. IgG *m/z* 1810.93

9.2.5.1 IDA

The spectra collected for IgG m/z 1810.93 are shown in Table 51. This sample yields a deep MS^n pathway, 1810.93_1551.79_1125.58_866.44_662.34_458.14, as shown in the first 6 entries of the table. Entry 7, 1810.93_1384.52, is selected because its composition has a scar on its reducing end only, and entry 8 follows as the MajorGlyco pathway from 7.

Table 51: Spectra suggested by IDA for IgG *m/z* 1810.93.

#	Spectrum Added	Relative Intensity	Putative Terminal Ion Compositions	Reason Added
1	**1810.93**	N/A	1810.93 H_3FN_3h 1810.93 H_4N_3f 1810.93 H_4FN_2n	Initial
2	1810.93_**1551.60**	100.0	1551.79 H_3FN_2h-(oh) 1551.79 H_4N2f-(oh) 1551.79 H_4FNn-(oh)	Major(1)
3	1810.93_1551.79_**1125.41**	100.0	1125.54 H_3N_2-(ene)(oh)' 1125.58 HFN_2h-(ene) 1125.58 H2N2f-(ene) 1125.58 H2FNn-(ene)	Major(2)
4	1810.93_1551.79_1125.58_**866.30**	100.0	866.40 H_3N-(ene)(oh)2' 866.44 HFNh-(ene)(oh) 866.44 H_2Nf-(ene)(oh) 866.44 H_2Fn-(ene)(oh)	Major(3)
5	1810.93_1551.79_1125.58_866.44_**662.23**	100.0	662.30 H_2N-(ene)$(oh)_2$' 662.34 FNh-(ene)(oh) 662.34 HNf-(ene)(oh) 662.34 HFn-(ene)(oh)	Major(4)
6	1810.93_1551.79_1125.58_866.44_662.34_**458.14**	100.0	458.20 HN-(ene)$(oh)_2$' 458.24 Nf-(ene)(oh) 458.24 Fn-(ene)(oh)	Major(5)
7	1810.93_**1384.52**	71.6	1384.68 H_3N_3-(ene)'	REScar(1)
8	1810.93_1384.60_**1125.40**	100.0	1125.54 H_3N_2-(ene)(oh)'	Major(7)

9.2.5.2 IsoSolve

Given these spectra and the **-NLinkedBranching** switch, IsoSolve returns 11 structures from 28 total pathways. See Table 52.

Table 52: IsoSolve results for the spectra shown in Table 51.

#	Score	Linear Code	Compatible Pathways	Cumulative
1	74.07	NH'(H')H'(H)N'(F)n'	20/27	71.43% (20/28)
2	74.07	NH'(HH')H'N'(F)n'	20/27	75.00% (21/28)
3	67.86	NHH'(H')H'N'(F)n'	19/28	78.57% (22/28)
4	64.29	NHH'(H')H'(F)N'n'	18/28	89.29% (25/28)
5	62.96	NH'(H')H'(FH)N'n'	17/27	89.29% (25/28)
6	59.26	N(F)H'(HH')H'N'n'	16/27	96.43% (27/28)
7	59.26	NH'(H')H'N'(F)(H)n'	16/27	100.00% (28/28)
8	59.26	NH'(H')H'N'(FH)n'	16/27	100.00% (28/28)
9	55.56	NH'(HH')H'(F)N'n'	15/27	100.00% (28/28)
10	44.44	N(F)H'(H')H'N'(H)n'	12/27	100.00% (28/28)
11	40.74	H'(H')H'N'(NHF)n'	11/27	100.00% (28/28)

9.2.5.3 Discussion

The expected structure—HNH'(H')H'N'(F)n—was not generated. After some investigation, it is discovered that the major pathway 1810.93_1551.79_1125.58_866.44_662.34_458.14_268.1 (which includes ion *m/z* 268.1 from spectrum entry 6 on Table 51) is not compatible with the *N*-linked core motif of H(H)HNn (see Figure 5 on page 10). This pathway, when provided as input to OSCAR, produces zero structures if the **-NLinkedBranching** switch is given.

Next we use the LabelPathway command to investigate possible compositions of the pathway's ions. Given the input of Listing 25 and the corresponding output in Listing 26, we see

that the only valid composition for the initial ion *m/z* 1810.93 is that it contains a reducing-end *reduced hexose* (h) instead of a *reduced HexNAc* (n). This finding is quite unexpected and demands further investigation.

```
-ReducingEndResidue nh
LabelPathway NoCrossRing 1810.93_1551.79_1125.58_866.44_662.34_458.14_268.1
```

Listing 25: OSCAR input to investigate the curious pathway 1810.93_1551.79_1125.58_866.44_662.34_458.14_268.1. The pathway came from an *N*-linked glycan, but is not consistent with a reducing-end n residue.

```
1810.93 H3FN3h
1551.79 H3FN2h-(oh)
1125.54 H3N2-(ene)(oh)'
 866.40 H3N-(ene)(oh)2'
 662.30 H2N-(ene)(oh)2'
 458.20 HN-(ene)(oh)2'
 268.12 N-(ene)(oh)'
```

Listing 26: Composition mapping for the ions from Listing 25. The initial ion of this pathway must contain a reduced hexose (h) instead of the expected reduced HexNAc (n).

Executing IsoSolve again on the input spectra, but this time specifying `-ReducingEndResidue h` but not `-NLinkedBranching` to better reflect what we know about this sample, we find 16 structures generated from 54 input pathways. At the top of the ranked list is the structure NH(NH)HN(F)h, with a score of 90.74%. This unusual structure is shown in Figure 47.

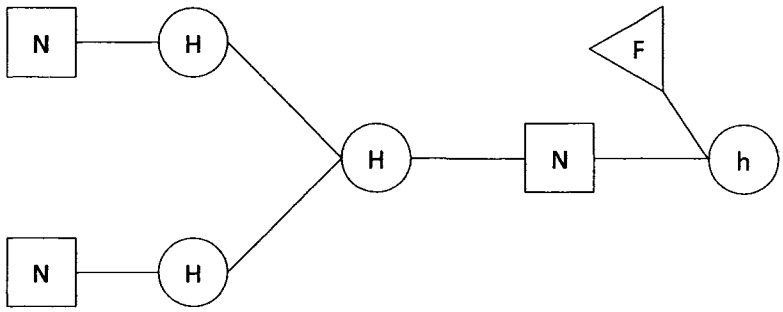


Figure 47: An *N*-linked glycan that does not contain the usual *N*-linked core motif H(H)HNn. Instead, this glycan from IgG *m/z* 1810.9 has a reducing-end reduced hexose (h).

This striking glycan does not contain the expected *N*-linked core motif and has apparently not been reported previously. Coincidentally, this same structure was recently identified in IgG *m/z* 1810 by committee member Dr. David Ashline, and is featured in a manuscript that is currently in preparation (Ashline 9). We should stress that this unreported and highly unusual glycan was identified by GlySpy with almost no direction from the analyst. In fact, the only decision made by the analyst was to change the options given to IsoSolve. The data collection and analysis was otherwise fully automated.

This stands as a strong example of why *de novo* analysis is central to GlySpy's philosophy. This structure would not have been discovered if GlySpy relied on databases of previously-reported glycans or on presumed biosynthetic constraints.

Recall also that the MS^n pathway that first hinted at this structure (1810.93_1551.79_1125.58_866.44_662.34_458.14_268.1) was the major glycosidic pathway, the series of *most abundant ions* on each successive spectrum. The glycan is not a low-abundance isomer hiding near the spectrum baselines. If anything, it may be the most abundant glycan at *m/z* 1810, and yet it has escaped detection by other methods.

9.2.6. IgG *m/z* 1851.96

9.2.6.1 IDA

Table 53 lists the spectra collected by IDA for IgG *m/z* 1851.96. As with previous structures, we see the MajorGlyco backbone formed in steps 1-6, followed by spectra selected because of the precursor scar pattern (step 7) or complementary nature (9). Step 8 is the MajorGlyco extension of step 7.

Table 53: Spectra suggested by IDA for IgG *m/z* 1851.96.

#	Spectrum Added	Relative Intensity	Putative Terminal Ion Compositions	Reason Added
1	**1851.96**	N/A	1851.96 H_2FN_4h 1851.96 H_3N_4f 1851.96 H_3FN_3n	Initial
2	1851.96_**1384.54**	100.0	1384.68 H_3N_3-(ene)'	Major(1)
3	1851.96_1384.68_**1125.42**	100.0	1125.54 H_3N_2-(ene)(oh)'	Major(2)
4	1851.96_1384.68_1125.55_**866.31**	100.0	866.40 H_3N-(ene)$(oh)_2$'	Major(3)
5	1851.96_1384.68_1125.55_866.40_**662.23**	100.0	662.30 H_2N-(ene)$(oh)_2$'	Major(4)
6	1851.96_1384.68_1125.55_866.40_662.30_**458.13**	100.0	458.20 HN-(ene)$(oh)_2$'	Major(5)
7	1851.96_**1592.63**	61.8	1592.81 H_2FN_3h-(oh) 1592.81 H_3N_3f-(oh) 1592.81 H_3FN_2n-(oh)	NREScar(1)
8	1851.96_1592.70_**1125.39**	100.0	1125.54 H_3N_2-(ene)(oh)' 1125.58 HFN_2h-(ene) 1125.58 H_2N_2f-(ene) 1125.58 H_2FNn-(ene)	Major(7)
9	1851.96_1592.70_**490.16**	3.0	490.23 HN-$(oh)_2$' 490.26 Nf-(oh) 490.26 Fn-(oh)	Complement(8)

9.2.6.2 IsoSolve

Given the spectra from Table 53, IsoSolve returns exactly one structure, the expected one, from 41 pathways. See Table 54.

Table 54: IsoSolve results for the spectra shown in Table 53.

#	Score	Linear Code	Compatible Pathways	Cumulative
1*	100.0	NH'(NH')H'N'(F)n'	41/41	100.00% (41/41)

9.2.6.3 Discussion

This result is surprising for its simplicity. As we have seen, many biologically-derived *N*-glycan samples appear to contain structural isomers, which have unexpected topologies. In this case, however, all 41 pathways are compatible with the single proposed topology.

9.2.7. Ovalbumin *m/z* 1187.61

9.2.7.1 IDA

The spectra collected by IDA for ovalbumin *m/z* 1187.6 are shown in Table 55 and reproduced as Spectrum A-37 through Spectrum A-45 beginning on page 228. Entry 6 (*m/z* 928.3) is notable and will be discussed below.

Table 55: Spectra suggested by IDA for Ovalbumin *m/z* 1187.61.

#	Spectrum Added	Relative Intensity	Putative Terminal Ion Compositions	Reason Added
1	**1187.61**	N/A	1187.61 H_2N_2h 1187.61 H_3Nn	Initial
2	1187.61_**894.32**	100.0	894.43 H_3N-(ene)'	Major(1)
3	1187.61_894.45_**676.23**	100.0	676.32 H_2N-(ene)(oh)'	Major(2)
4	1187.61_894.45_676.36_**431.10**	100.0	431.19 H_2-(ene)(oh)'	Major(3)
5	1187.61_894.45_**667.24**	21.0	667.32 H_3-(oh)'	REScar(2)
6	1187.61_**928.34**	2.2	928.47 H_2Nh-(oh) 928.47 H_3n-(oh)	NREScar(2)
7	1187.61_928.34_**724.28**	100.0	724.37 HNh-(oh) 724.37 H_2n-(oh)	Major(6)
8	1187.61_894.45_667.24_**449.12**	100.0	449.20 H_2-$(oh)_2$'	Major(5)
9	1187.61_928.34_724.28_**506.27**	100.0	506.26 Nh-$(oh)_2$ 506.26 Hn-$(oh)_2$	Major(7)

9.2.7.2 IsoSolve

Given the spectra of Table 55, IsoSolve returns 10 structures from 49 total pathways. (See Table 56.) IsoSolve in this instance was executed *without* the –NLinkedBranching switch, and so its output is not limited to the usual *N*-linked core structure. The reducing-end residues were allowed to be either h or n.

Table 56: IsoSolve results for the spectra shown in Table 55.

#	Score	Linear Code	Compatible Pathways	Cumulative
1*	89.29	H(H)HNn	25/ 28	51.02% (25/49)
2	59.46	NH(H)Hn	22/ 37	83.67% (41/49)
3	57.14	HHHNn	16/ 28	87.76% (43/49)
4	43.48	NH(N)Hh	10/ 23	97.96% (48/49)
5	41.67	NH(N)(H)h	10/ 24	97.96% (48/49)
6	41.67	NH(H)Nh	10/ 24	97.96% (48/49)
7	41.38	N(H)(H)Hn	12/ 29	97.96% (48/49)
8	33.33	NH(HH)n	8/ 24	100.00% (49/49)
9	30.43	NHH(N)h	7/ 23	100.00% (49/49)
10	30.43	NHHNh	7/ 23	100.00% (49/49)

9.2.7.3 Discussion

The *N*-glycan isolated from ovalbumin *m/z* 1187.6 is widely assumed to be the usual five-residue core H_3Nn. Even without the –NLinkedBranching switch, IsoSolve reported this as, by far, the highest-scoring structure. The structure is well-covered by spectrum entries 1-5 and 8. However, it is interesting that this structure was not compatible with all of the pathways gathered. Might a structural isomer be lurking even here?

A glance at IsoSolve's output reveals that the 1187.61_928.34 pathway of Table 55 is not compatible with the expected core structure. Spectrum A-45 is the final spectrum in this MajorGlyco pathway, and reveals only one glycosidic fragment, *m/z* 316.2. Executing the command `LabelPathway NoCrossRing 1187.61_928.34_724.28_506.27_316.18` reveals the unambiguous composition pathway of Table 57.

Table 57: The composition of the anomalous pathway from Table 56.

#	*m/z*	Composition
1	1187.61	H_3Nn
2	928.47	H_3n-(oh)
3	724.37	H_2n-(oh)
4	506.26	Hn-$(oh)_2$
5	316.17	n-(oh)

This pathway, when provide to OSCAR via the AddPathway command, produces a single possible structure: NH(H)Hn. This is the second highest-scoring structure of Table 56. Again it appears that GlySpy has discovered an unexpected isomeric glycan without human intervention and, again, biosynthetic rules would certainly have disallowed this structural assignment.

An important point here is the low abundance of this isomer. As entry 6 in Table 55 shows, the *m/z* 928 product has a relative intensity of only 2.2%. Clearly the analysis of this isomer benefits from MS^n's ability to isolate low-abundance structures by selecting ions that cannot be generated by the higher-abundance glycans. This analytical specificity is unique to MS^n.

Lastly, the isomer proposed here, NH(H)Hn, bears obvious similarities to the unusual isomers proposed in Section 9.2.3.3, in particular the reducing end motif Hn. It is possible that these structures are exemplars of a new class of *N*-linked core structures.

9.2.8. Ovalbumin *m/z* 1636.84

9.2.8.1 IDA

Table 58: Spectra suggested by IDA for Ovalbumin *m/z* 1636.84.

#	Spectrum Added	Relative Intensity	Putative Terminal Ion Compositions	Reason Added
1	**1636.80**	N/A	1636.84 H_3N_3h 1636.84 H_4N_2n	Initial
2	1636.80_**1343.51**	100.0	1343.66 H_4N_2-(ene)'	Major(1)
3	1636.80_1343.60_**1084.40**	100.0	1084.52 H_4N-(ene)(oh)'	Major(2)
4	1636.80_1343.60_1084.50_ **866.32**	100.0	866.40 H_3N-(ene)(oh)2'	Major(3)
5	1636.80_1343.60_1084.50_ 866.40_**621.21**	88.6	621.27 H_3-(ene)(oh)2'	Major(4)
6	1636.80_1343.60_1084.50_ 866.40_**463.12**	39.4	463.22 H_2-(oh)'	REScar(5)
7	1636.80_**1377.57**	35.9	1377.70 H_3N_2h-(oh) 1377.70 H_4Nn-(oh)	NREScar(1)
8	1636.80_1343.60_1084.50_**463.20**	4.3	463.22 H_2-(oh)'	REScar(3)
9	1636.80_1343.60_**1116.44**	4.2	1116.54 H_4N-(oh)'	REScar(2)
10	1636.84_1377.57_**1084.39**	100.0	1084.52 H_4N-(ene)(oh)'	Major(7)
11	1636.84_1377.57_**1118.45**	10.6	1118.56 H_3Nh-(oh)2 1118.56 H_4n-(oh)2	NREScar(7)

Surprisingly, spectrum 5 in this table is the major pathway from its precursor, yet it has a relative intensity of only 88.6% instead of the expected 100%. This is because the major peak on that spectrum is *m/z* 709.36, which represents a cross-ring cleavage of the exposed reducing-end HexNAc. This is an example where naïve, intensity-driven data collection might select the less desirable *m/z* 709.36 instead of *m/z* 621.21 as was done here.

Also, pathway 1636.80 → 1343.60 → 1084.50 → 866.40 → 621.27 → 463.36 was suggested by IDA for spectrum 6. However, we were unable to acquire the spectrum due to instrument sensitivity limitations, and so we used the `DoNotSuggestPathway` command to prevent it

from being suggested again. Interestingly, the next pathway suggested (number 6 in the table) was the same terminal ion, but without the intermediate ion *m/z* 621.27.

9.2.8.2 IsoSolve

IsoSolve returned 12 structures from 67 total pathways.

Table 59: IsoSolve results for the spectra shown in Table 58.

#	Score	Linear Code	Compatible Pathways	Cumulative
1	77.61	HH'(N)(H')H'N'n'	77.61% (52/67)	52/67
2	73.44	NHH'(H')H'N'n'	94.03% (63/67)	47/64
3	71.88	NH'(H)(H')H'N'n'	94.03% (63/67)	46/64
4	71.88	NH(H')(H')H'N'n'	94.03% (63/67)	46/64
5	70.31	N(H)H'(H')H'N'n'	95.52% (64/67)	45/64
6*	68.18	NH'(HH')H'N'n'	95.52% (64/67)	45/66
7	54.39	NH'(H')H'(H)N'n'	97.01% (65/67)	31/57
8	53.33	HH'(H')H'(N)N'n'	97.01% (65/67)	32/60
9	53.33	H'(H')(H)H'N'(N)n'	100.00% (67/67)	8/15
10	53.33	H'(H')H'(H)N'(N)n'	100.00% (67/67)	8/15
11	53.33	HH'(H')H'N'(N)n'	100.00% (67/67)	8/15
12	50.88	H'(H')H'(NH)N'n'	100.00% (67/67)	29/57

9.2.8.3 Discussion

The expected structure appears as entry 6 of Table 59, but the other suggested structures have merit as well. For example, entry 1, HH'(N)(H')H'N'n' contains a bisecting HexNAc while the expected structure does not. Evidence for both structures is present. In Spectrum A-46, and in the detail shown in Spectrum A-47, we see that the core structure generates both *m/z* 444 and *m/z* 458 fragments. (See page 233.) As we have seen before (Section 9.2.4.3), these ions are

diagnostic of structures with and without a bisecting HexNAc, respectively. Again GlySpy has apparently discovered a structural isomer that was unreported in (Harvey 37), and has done so with no human intervention. Future improvements to GlySpy could add improved facile-cleavage scoring to further narrow the candidates of Table 59, but even today the software has produced a candidate list that can be easily managed by an analyst.

9.2.9. Ovalbumin *m/z* 1677.87

9.2.9.1 IDA with spectrum pruning disabled and enabled

Table 60 shows the spectra collected for ovalbumin *m/z* 1677.8 with spectrum pruning *disabled*. Table 61 shows the spectra collected spectrum pruning *enabled*.

Table 60: Spectra suggested by IDA for ovalbumin *m/z* 1677.87 with pruning disabled.

#	Spectrum Added	Relative Intensity	Putative Terminal Ion Compositions	Reason Added
1	**1677.70**	N/A	1677.86 F_4S_2f 1677.87 H_2N_4h 1677.87 H_3N_3n	Initial
2	1677.70_**1384.50**	29.2	1384.68 H_3N_3-(ene)'	Major(1)
3	1677.70_1384.60_**1125.38**	100.0	1125.54 H_3N_2-(ene)(oh)'	Major(2)
4	1677.70_1384.60_1125.50_**866.45**	100.0	866.40 H_3N-(ene)$(oh)_2$'	Major(3)
5	1677.70_1384.60_1125.50_866.50_**662.41**	100.0	662.30 H_2N-(ene)$(oh)_2$'	Major(4)
6	1677.87_1384.60_1125.50_866.40_662.40_**458.11**	100.0	458.20 HN-(ene)$(oh)_2$'	Major(5)
7	1677.70_**1418.54**	18.7	1418.73 H_2N_3h-(oh) 1418.73 H_3N_2n-(oh)	NREScar(1)
8	1677.87_1418.54_**1125.39**	100.0	1125.54 H_3N_2-(ene)(oh)'	Major(7)
9	1677.87_1418.54_1125.39_**866.30**	100.0	866.40 H_3N-(ene)$(oh)_2$'	Major(8)
10	1677.87_1418.54_1125.39_866.30_**662.22**	100.0	662.30 H_2N-(ene)$(oh)_2$'	Major(9)
11	1677.87_1418.54_1125.39_866.30_662.22_**458.15**	100.0	458.20 HN-(ene)$(oh)_2$'	Major(10)
12	1677.87_1418.54_**1159.45**	20.7	1159.58 H_2N_2h-$(oh)_2$ 1159.58 H_3Nn-$(oh)_2$	NREScar(7)
13	1677.87_1418.54_1159.45_**866.30**	100.0	866.40 H_3N-(ene)$(oh)_2$'	Major(12)
14	1677.87_1418.54_1159.45_866.30_**662.24**	100.0	662.30 H_2N-(ene)$(oh)_2$'	Major(13)
15	1677.87_1418.54_1159.45_866.30_662.24_**458.11**	100.0	458.20 HN-(ene)$(oh)_2$'	Major(14)
16	1677.87_1418.54_1159.45_**900.36**	15.9	900.44 H_2Nh-$(oh)_3$ 900.44 H_3n-$(oh)_3$	NREScar(12)
17	1677.87_1418.54_1159.45_900.36_**710.25**	100.0	710.36 HNh-$(oh)_2$ 710.36 H_2n-$(oh)_2$	Major(16)

Table 61: Spectra suggested by IDA for ovalbumin *m/z* 1677.87 with pruning enabled.

#	Spectrum Added	Relative Intensity	Putative Terminal Ion Compositions	Reason Added
1	**1677.70**	N/A	1677.86 F_4S_2f 1677.87 H_2N_4h 1677.87 H_3N_3n	Initial
2	1677.70_**1384.50**	29.2	1384.68 H_3N_3-(ene)'	Major(1)
3	1677.70_1384.60_**1125.38**	100.0	1125.54 H_3N_2-(ene)(oh)'	Major(2)
4	1677.70_1384.60_1125.50_**866.45**	100.0	866.40 H_3N-(ene)$(oh)_2$'	Major(3)
5	1677.70_1384.60_1125.50_866.50_**662.41**	100.0	662.30 H_2N-(ene)$(oh)_2$'	Major(4)
6	1677.87_1384.60_1125.50_866.40_662.40_**458.11**	100.0	458.20 HN-(ene)$(oh)_2$'	Major(5)
7	1677.70_**1418.54**	18.7	1418.73 H_2N_3h-(oh) 1418.73 H_3N_2n-(oh)	NREScar(1)
8	1677.87_1418.54_**1125.39**	100.0	1125.54 H_3N_2-(ene)(oh)'	Major(7)
9	1677.87_1418.54_**1159.45**	20.7	1159.58 H_2N_2h-$(oh)_2$ 1159.58 H_3Nn-$(oh)_2$	NREScar(7)
10	1677.87_1418.54_1159.45_**866.30**	100.0	866.40 H_3N-(ene)$(oh)_2$'	Major(9)
11	1677.87_1418.54_1159.45_**900.36**	15.9	900.44 H_2Nh-$(oh)_3$ 900.44 H_3n-$(oh)_3$	NREScar(10)
12	1677.87_1418.54_1159.45_900.36_**710.25**	100.0	710.36 HNh-$(oh)_2$ 710.36 H_2n-$(oh)_2$	Major(11)

The obvious difference between these tables is the number of spectra collected: 17 with pruning disabled versus 12 with it enabled. As you can see, entries 9-11 and 14-15 of Table 60 show the re-elaboration of the *m/z* 866.3 → 662.2 → 458.1 MajorGlyco pathways. These additional spectra add little or no useful information. See Spectrum A-48, Spectrum A-49, and Spectrum A-50 beginning on page 234 to see the obvious similarities between the *m/z* 662.2 spectra (table entries 5, 10, and 14, respectively). Pruning in this case succeeds in rejecting structurally uninformative spectra, reducing data collection time.

The following analysis continues using the data from Table 61 (pruning enabled).

Notice that entry 2 this table lists 1677.70_1384.50 as the MajorGlyco pathway, and yet it only has a relative intensity of 29.2%. A close examination of Spectrum A-51 begins to reveal the problem. The major peak is labeled 1417.5, and apparently represents the loss of a terminal

HexNAc, yielding a composition of H_3N_2n-(oh). However, that composition has a theoretical *m/z* of 1418.73, which is 1.23 mass units and away from the observed value, outside the default mass error window of ±0.5 Da. This is an error of 867 ppm, which is quite large for the LTQ. The logical conclusion is that these data were collected when the mass spectrometer was in a poorly calibrated state.

The ion *m/z* 1384.50 (entry 2) is also incorrect by a similar amount. The exact composition is H_3N_3-(ene)', *m/z* 1384.68, but Spectrum A-51 shows the measured *m/z* as 1383.5, an error of 1.18 mass units or 852 ppm. So where did the ion *m/z* 1384.50 come from, if the observed value is 1383.5? It is the +1 peak of the isotopic envelope. Spectrum A-52 presents a magnified view of the spectrum and clearly shows the *m/z* 1384.5 (+1) peak that is selected by IDA. Notice also that the *m/z* 1418.6 (+1) peak is present, but at a lower abundance than *m/z* 1384.5. Since IDA chooses only the most intense glycosidic peak, *m/z* 1384.5 is selected.

Ions subsequently selected and fragmented from this +1 peak are remarkably close to their theoretical values, as the extra +1 mass unit in the precursor roughly offsets the -1 amu calibration error.

This leads to several conclusions about automated data collection and mass accuracy:

1) The current method is naïve about measurement error and should be improved. Future work could include automatically calibrating the spectrum or reporting unusually large suspected errors.

2) However, in this case, the data collected are still usable and, as we will see, lead to appropriate analytical conclusions.

3) In this case, though, the major pathway should have begun with *m/z* 1418.73, which is far more intense than the +1 ion *m/z* 1384.5 that was actually selected. This clearly limits the maximum depth of the MS^n experiment.

9.2.9.2 IsoSolve

When given the spectra from Table 61, IsoSolve returns five structures from 61 total pathways. See Table 62.

Table 62: IsoSolve results for the spectra shown in Table 61.

#	Score	Linear Code	Compatible Pathways	Cumulative
1	88.52	NH'(NH')H'N'n'	54/ 61	88.52% (54/61)
2*	88.14	NH'(N)(H')H'N'n'	52/ 59	96.72% (59/61)
3	78.85	N(N)H'(H')H'N'n'	41/ 52	98.36% (60/61)
4	57.63	NH'(H')H'(N)N'n'	34/ 59	98.36% (60/61)
5	34.62	H'(H')H'(N)(N)N'n'	18/ 52	100.00% (61/61)

9.2.9.3 Discussion

The expected structure ranks second in the output list, narrowly behind the top candidate. The topological difference between these structures is quite simple: structure 1 has terminal HexNAcs on each antenna, whereas structure 2 moves one of these Ns to become a bisecting HexNAc. The evidence strongly suggests that both structures are present. Spectrum A-49 shows the expected fragments for structures with and without a bisecting HexNAc: *m/z* 444.1 (not labeled, composition HN-(ene)$(oh)_3$') and *m/z* 458.3 (composition HN-(ene)$(oh)_2$'). We conclude that GlySpy has again identified a structural isomer that was invisible to other analytical techniques.

9.2.10. Ovalbumin *m/z* 1922.99

9.2.10.1 IDA

Table 63 shows the spectra collected by IDA for ovalbumin *m/z* 1922.99. Again we see the MajorGlyco pathway extended (steps 1-7), reducing-end and non-reducing-end scar fragments selected (and 8, 9, 12, and 14), and those scar fragments themselves being extended by their major ions (steps 10, 11, and 13). A total of 14 spectra were collected for a glycan containing eight residues.

Table 63: Spectra suggested by IDA for ovalbumin *m/z* 1922.99.

#	Spectrum Added	Relative Intensity	Putative Terminal Ion Compositions	Reason Added
1	**1923.00**	N/A	1922.98 F_4NS_2f 1922.99 H_2N_5h 1922.99 H_3N_4n	Initial
2	1923.00_**1663.59**	100.0	1663.84 F_4S_2f-(oh) 1663.85 H_2N_4h-(oh) 1663.85 H_3N_3n-(oh)	Major(1)
3	1923.00_1663.70_**1370.46**	100.0	1370.67 H_3N_3-(ene)(oh)'	Major(2)
4	1923.00_1663.70_1370.44_ **1111.40**	100.0	1111.53 H_3N_2-(ene)$(oh)_2$'	Major(3)
5	1923.00_1663.70_1370.44_ 1111.40_**852.28**	100.0	852.38 H_3N-(ene)$(oh)_3$'	Major(4)
6	1922.99_1663.59_1370.46_ 1111.40_852.28_**662.33**	100.0	662.30 H_2N-(ene)$(oh)_2$'	Major(5)
7	1922.99_1663.59_1370.46_ 1111.40_852.28_**634.26**	100.0	634.27 H_2N-(ene)$(oh)_4$'	Major(5)
8	1923.00_1663.70_**1404.48**	91.0	1404.71 H_2N_3h-$(oh)_2$ 1404.71 H_3N_2n-$(oh)_2$	NREScar(2)
9	1923.00_**1629.54**	44.5	1629.81 H_3N_4-(ene)'	REScar(1)
10	1923.00_1663.70_1404.60_ **1111.37**	100.0	1111.53 H_3N_2-(ene)$(oh)_2$'	Major(8)
11	1923.00_1629.64_**1370.44**	100.0	1370.67 H_3N_3-(ene)(oh)'	Major(9)
12	1923.00_1663.70_1404.60_ **1145.38**	46.6	1145.57 H_2N_2h-$(oh)_3$ 1145.57 H_3Nn-$(oh)_3$	NREScar(8)
13	1923.00_1663.70_1404.60_ 1145.45_**852.28**	100.0	852.38 H_3N-(ene)$(oh)_3$'	Major(12)
14	1923.00_1663.70_1404.60_ 1145.45_**886.29**	31.9	886.43 H_2Nh-$(oh)_4$ 886.43 H_3n-$(oh)_4$	NREScar(12)

Note that entries 6 and 7 are both MajorGlyco extensions of entry 5. As seen before, spectrum 5 contained two different scans; on one, *m/z* 662.33 was the base peak, and on the other, *m/z* 634.26 was.

9.2.10.2 IsoSolve

Given the spectra of Table 63, IsoSolve returns five structures from 74 total pathways.

Table 64: IsoSolve results for the spectra shown in Table 63.

#	Score	Linear Code	Compatible Pathways	Cumulative
1*	90.54	N(N)H'(N)(H')H'N'n'	67/74	90.54% (67/74)
2	87.67	NH'(N)(N)(H')H'N'n'	64/73	98.65% (73/74)
3	76.71	N(N)(N)H'(H')H'N'n'	56/73	100.00% (74/74)
4	76.71	N(N)H'(NH')H'N'n'	56/73	100.00% (74/74)
5*	73.61	NH'(NH')(N)H'N'n'	53/72	100.00% (74/74)

9.2.10.3 Discussion

There are two expected structures at this mass, and they appear in Table 64 and Figure 48 as structures number 1 and 5.

Structures 2 and 3 are likely to be the most controversial; 2 proposes a rare double-bisecting HexNAc and 3 presents a triply-substituted hexose. What is the quality of evidence for these structures? To begin, we turn our attention to Spectrum A-53 (pathway 1922.99_1663.59_1370.46_1111.40_852.28_634.26; see page 237), which matches entry 7 of Table 63. The precursor ion *m/z* 634.3 has composition H_2N-(ene)$(oh)_4$', indicating that some subtree of two hexoses and one HexNAc had a total of four non-reducing-end scars and one reducing-end scar. Structures 4 and 5 cannot generate this fragment and so are excluded from this analysis. Table 65 shows the compositions of the product ions found on that spectrum, and maps those compositions to specific residues within each structure. If no mapping is possible, the table entry is left blank.

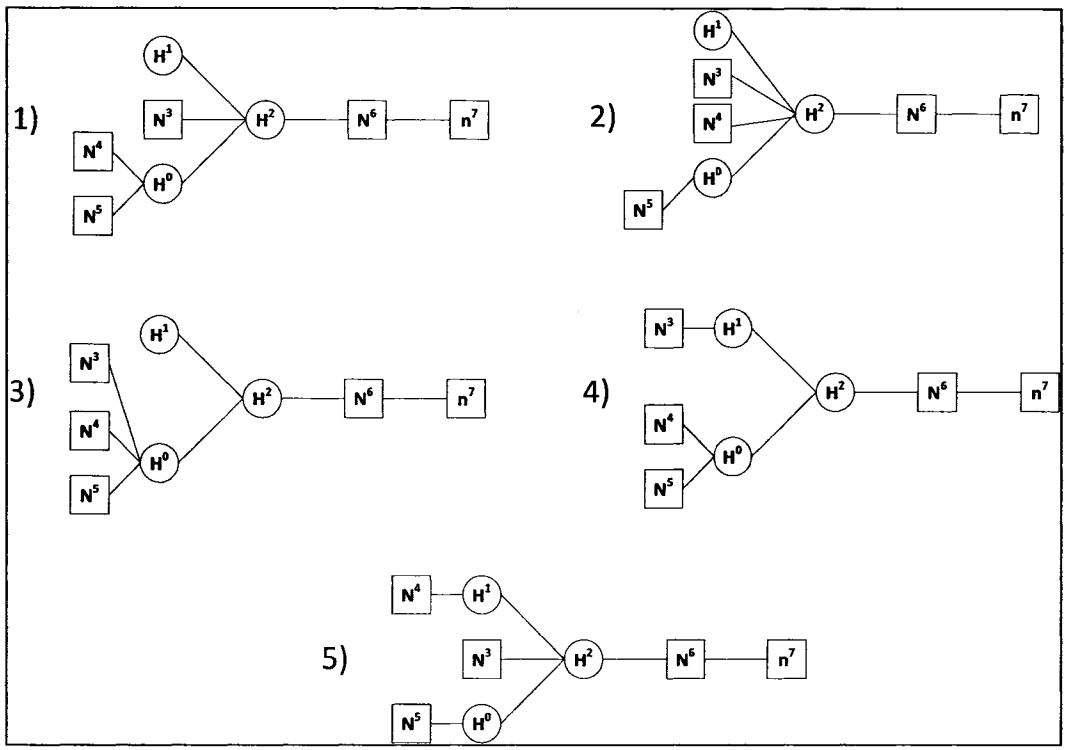


Figure 48: Structures 1-5 from Table 64.

Table 65: Putative compositions for observed ions on the ovalbumin spectrum *m/z* 1922.99_1663.59_1370.46_1111.40_852.28_634.26 and their mapping to structures 1, 2, and 3 of Table 64.

Ion *m/z*	Theoretical *m/z* and Composition	Corresponding Residues		
		Structure 1	Structure 2	Structure 3
634.3	634.27 H_2N-(ene)$(oh)_4$'	$H^0H^2N^6$	$H^0H^2N^6$	$H^0H^2N^6$
213.0	213.07 H-(ene)$(oh)_2$'	H^0		
231.1	231.08 H-$(oh)_3$'	H^0		
268.0	268.12 N-(ene)(oh)'	N^6	N^6	N^6
389.2	389.14 H_2-(ene)$(oh)_4$'	H^0H^2	H^0H^2	H^0H^2
407.2	407.15 H_2-$(oh)_5$'	H^0H^2	H^0H^2	H^0H^2
430.2	430.17 HN-(ene)$(oh)_4$'		H^2N^6	
444.3	444.18 HN-(ene)$(oh)_3$'	H^2N^6		

Table 65 reveals that the only ion that exclusively supports structure 2 is *m/z* 430.2; all other ions also support structure 1 and/or 3. Unfortunately, the spectrum acquired for *m/z* 430.2 falls well below an acceptable normalization level and we are therefore left with insufficient evidence to conclude that structure 2 is correct.

Structure 3 is interesting because OSCAR requires only a single pathway to generate it: *m/z* 1923.00_1663.70_1404.60_1145.45_969.38. Because multiple pathways are not used, the possibility of mixing pathways from multiple isomers is excluded. GlySpy's **`LabelPathway`** command gives composition assignments to these ions as shown in Table 66.

Table 66: Putative ion compositions for the pathway that supports structure 3 of Table 64.

Theoretical *m/z*	Composition
1922.99	H_3N_4n
1663.85	H_3N_3n-(oh)
1404.71	H_3N2n-$(oh)_2$
1145.57	H_3Nn-$(oh)_3$
969.50	H_2Nn-(oh)

As you can see, ion *m/z* 969.5 requires that a subtree of composition H_2Nn-(oh) can be isolated with a single non-reducing-end cleavage. In other words, one H and three Ns can be lost with a single cleavage. Of the structures in Table 64, only structure 3 can satisfy this, using the residues $H^0H^2N^6n^7$. Again, however, attempts to acquire a reliable spectrum from the pathway 1923.00_1663.70_1404.60_1145.45_969.5 failed due to instrument sensitivity limits.

CHAPTER 10:

SUMMARY AND CONTRIBUTIONS

This work has presented GlySpy, a suite of tools used to assign glycan structures from sequential mass spectral data. The document introduced the background material required to place GlySpy in context and reviewed a number of tools, techniques, and databases in the growing field of glycomics. It described the four main components of GlySpy, detailing the algorithms, providing experimental data, and presenting results. To summarize, these tools are:

- **OSCAR** (the Oligosaccharide Subtree Constraint Algorithm), which accepts MS^n disassembly pathways and produces a set of plausible glycan structures;
- **IsoDetect**, which reports the MS^n disassembly pathways that are inconsistent with a set of expected structures, and which therefore may indicate the presence of alternative isomeric structures;
- **IsoSolve**, which assigns the branching structures of multiple isomeric glycans found in a complex mixture; and
- **Intelligent Data Acquisition** (IDA), which provides automated guidance to the mass spectrometer operator, selecting glycan fragments for further MS^n disassembly.

GlySpy's contributions to the fields of glycomics and computer science include:

1) ***De novo* analysis.** GlySpy does not use presumed biosynthetic rules or previously-reported glycans to guide its analysis. Because of this *de novo* approach, GlySpy has assigned novel structures. For example, this document has presented evidence for *N*-glycans that do not contain the expected H_3Nn core, structures that have been overlooked by previous research. Unexpected structures such as these will lead to a deeper understanding of glycan synthesis and function. As glycan biosynthesis is perturbed by a variety of disease processes, similar unorthodox structures may someday be identified as important biomarkers.

2) **Performance.** Careful software engineering was applied to many portions of GlySpy to avoid the combinatorial problems commonly associated with glycan analysis. Examples include:

    a. OSCAR's fork data structure allows it to eliminate candidate structures without first constructing them.

    b. OSCAR optimizes composition pathways based on precursor/product relationships, greatly reducing the number of interpretations to be analyzed.

    c. OSCAR discards inconsistent and isomorphic forks, focusing its efforts on those forks whose interpretations can lead to unique assignments.

    d. Although GlySpy's analysis is *de novo*, a few important options are provided to allow analysts to restrict the search space, for example, the **`AddPathway`** command's **`NoCrossRing`** option, which eliminates cross-ring compositions from consideration. Also important are the **`-NLinked`** and

`-NLinkedBranching` options, which restrict analysis to structures that embed the H_3Nn core.

e. IsoSolve applies innovative techniques to guide the search for isomeric structures, despite the daunting size of the search space.

3) **Interpretation of higher-order MS^n data.** Many existing tools rely primarily or even exclusively on MS^2 spectra. As we have demonstrated, structural isomers may be hidden at this level of analysis, and higher-order analysis is required to find them. Additionally, although previous tools such as STAT have recognized that glycan fragments can be represented as subtrees, GlySpy extends this analysis to arbitrary levels of glycan disassembly, and also takes advantage of the precursor/product relationship at each level to reduce the number of candidate structures generated.

4) **Automation.** GlySpy's Intelligent Data Acquisition module presents one plausible model for automated data acquisition. When coupled with methods for directly controlling the mass spectrometer, GlySpy will offer fully automated data acquisition, greatly increasing instrument throughput. When this capability, in turn, is combined with IsoSolve, analytical throughput will be dramatically improved.

5) **General solution.** Nearly all of OSCAR's inference rules operate on the tree abstraction of glycans and encode no chemistry knowledge whatsoever. The remaining rules are based on cross-ring cleavages, which are perhaps specific to this domain. If a problem space were discovered that had similar characteristics—especially the notion of disassembling a tree structure in a way that leaves visible scars—then OSCAR's techniques may applicable. Further, depending upon the details of this new problem space, algorithms such as IsoDetect, IsoSolve, and

Intelligent Data Acquisition may also be appropriate. APPENDIX B: SAMPLE OSCAR INFERENCE RULES provides details of many of these rules and indicates how they may be applicable to a wider range of tree-based problems.

6) **Research aid.** GlySpy's demonstrated high performance and progress toward automated glycan analysis position it as the computational engine of a future high-throughput glycomics platform. Such a platform could be of value to many biological and medical researchers.

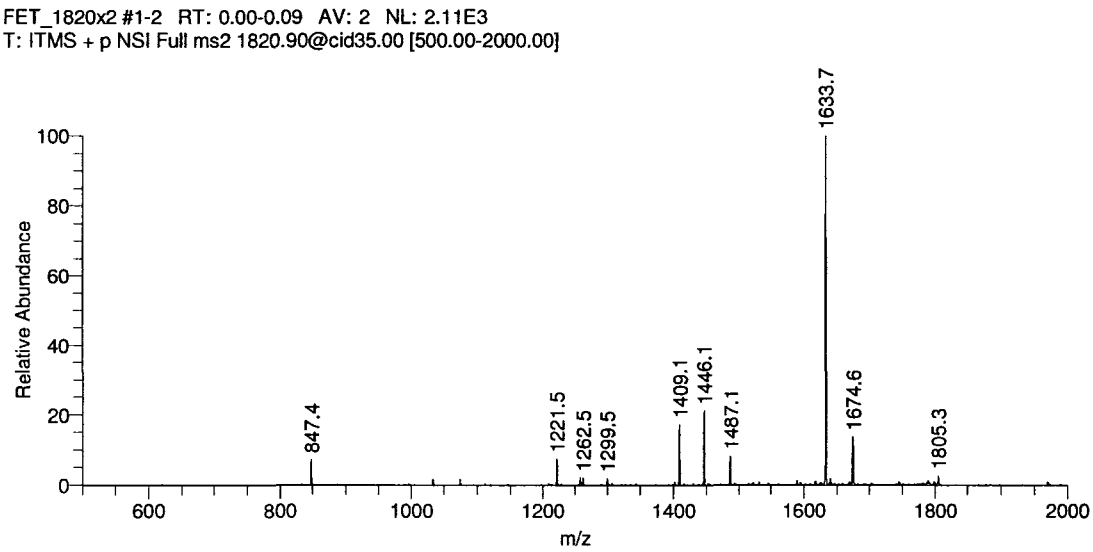
Although much work remains to be done in the field of glycomics, it is hoped that GlySpy has made a contribution to the fascinating problem of glycan structural analysis.

APPENDICES

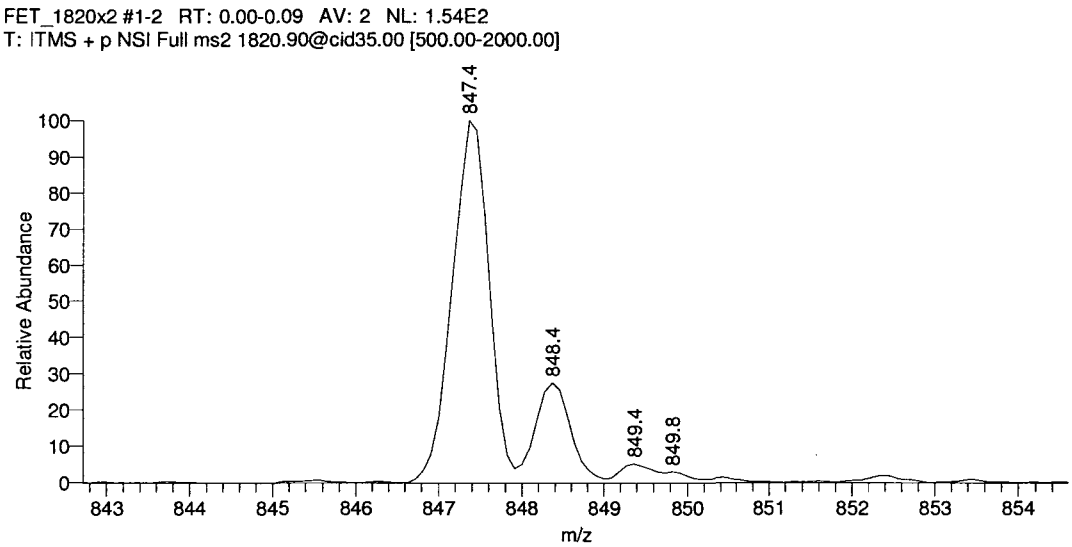
APPENDIX A:

SELECTED EXPERIMENTAL SPECTRA

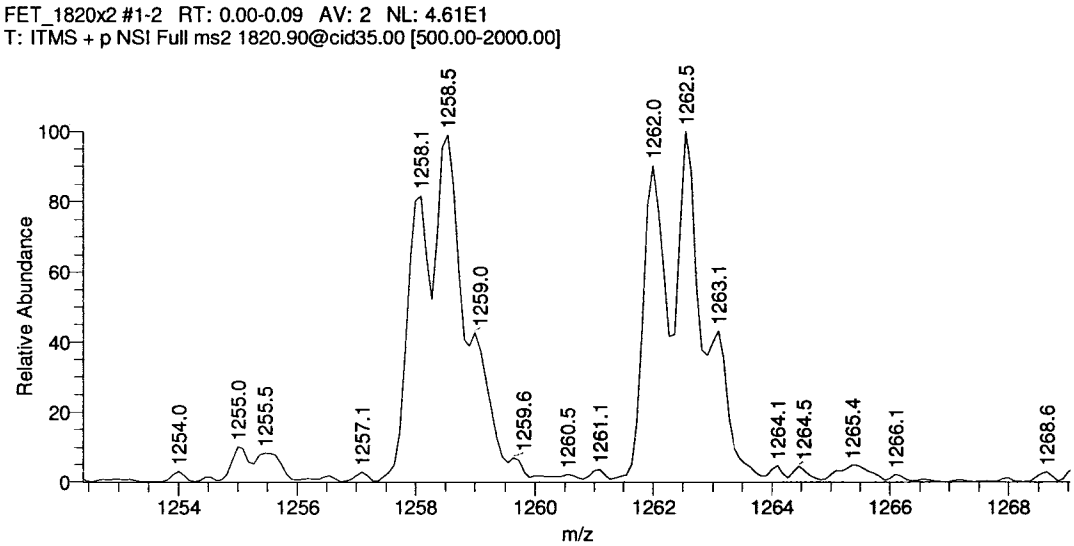
A.1. Fetuin Spectra



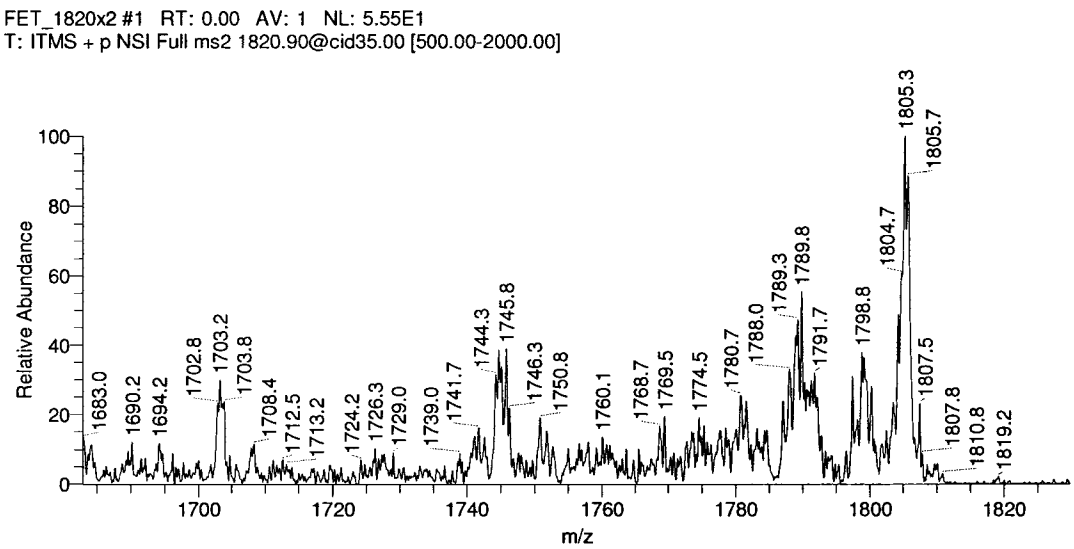
Spectrum A-1: Fetuin *m/z* 1820.9^{2+}



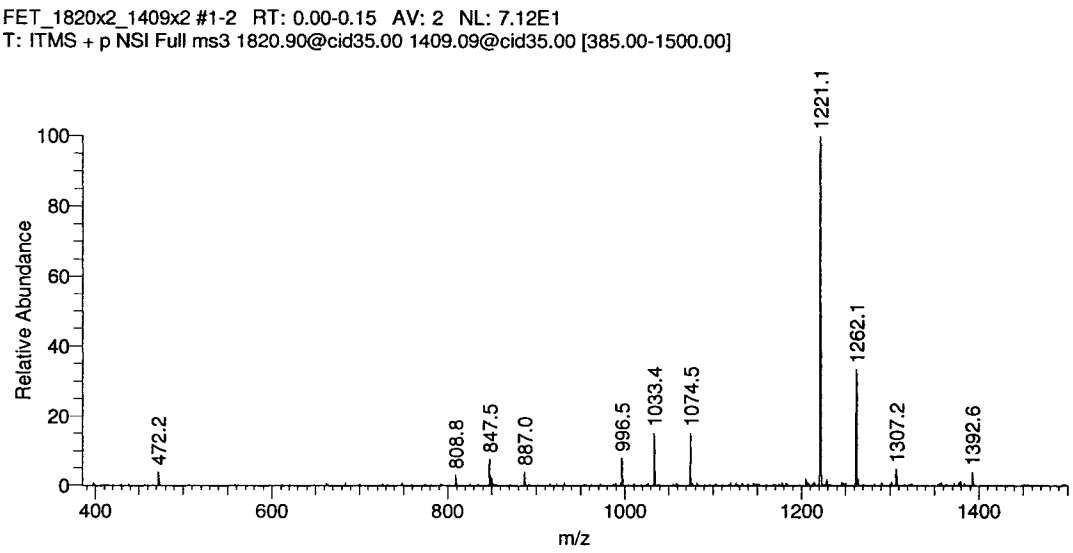
Spectrum A-2: Detail of *m/z* 874.4 reveals the charge state as +1.



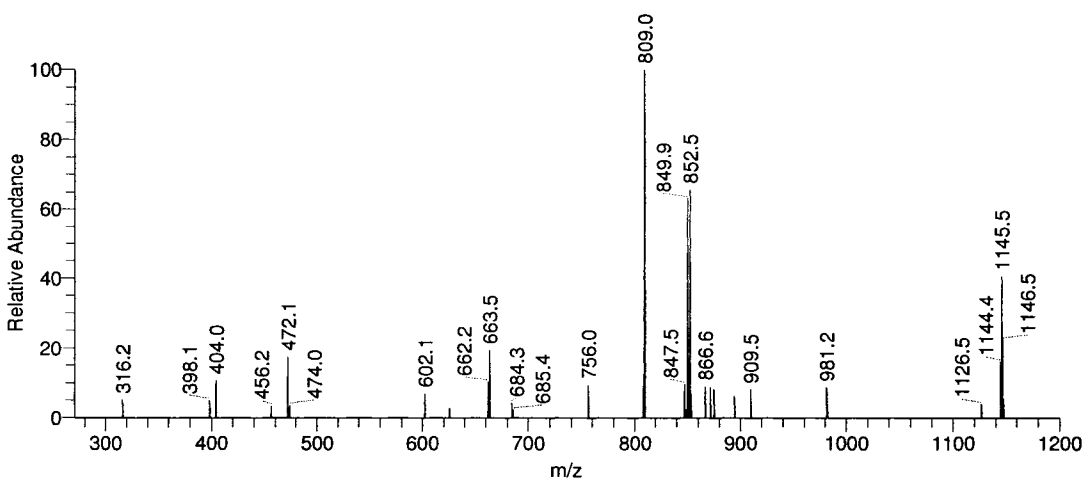
Spectrum A-3: Detail of ions *m/z* 1258.1 and *m/z* 1262.0 reveals both charge states as +2.



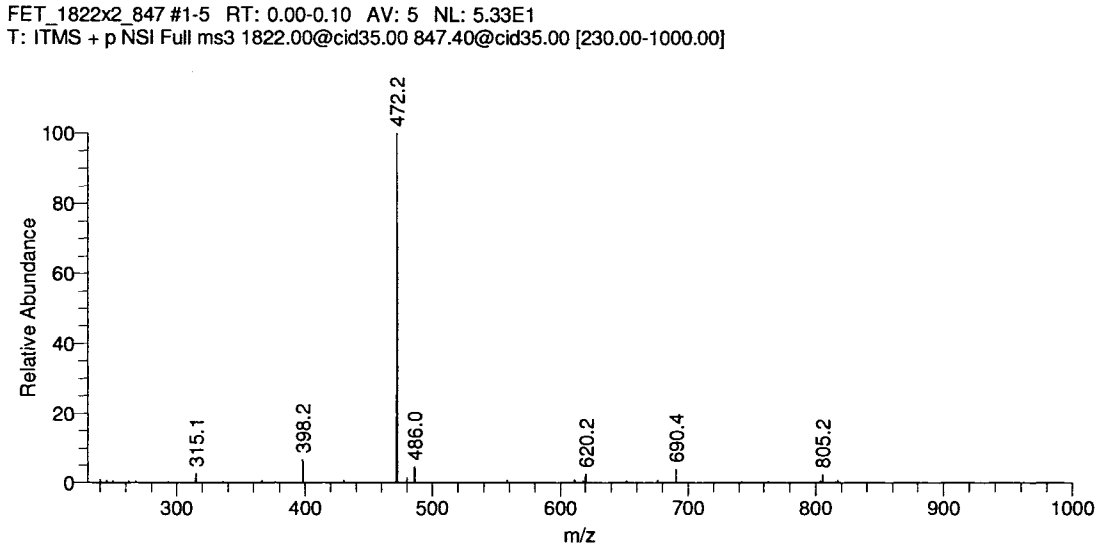
Spectrum A-4: Detail of presumptive electronic noise in the high *m/z* range.



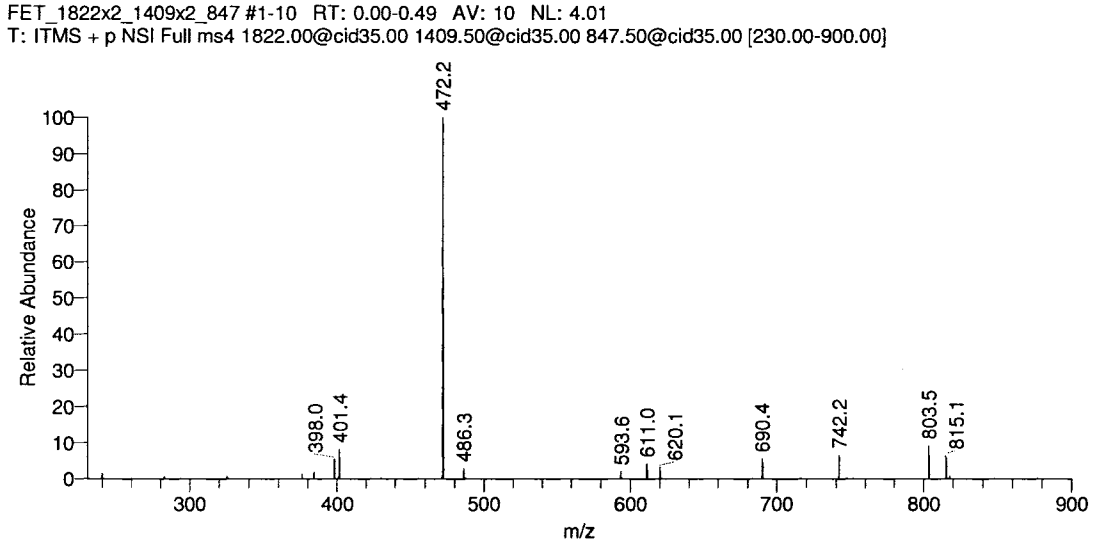
Spectrum A-5: Fetuin *m/z* $1820.9^{2+} \rightarrow 1408.5^{2+}$



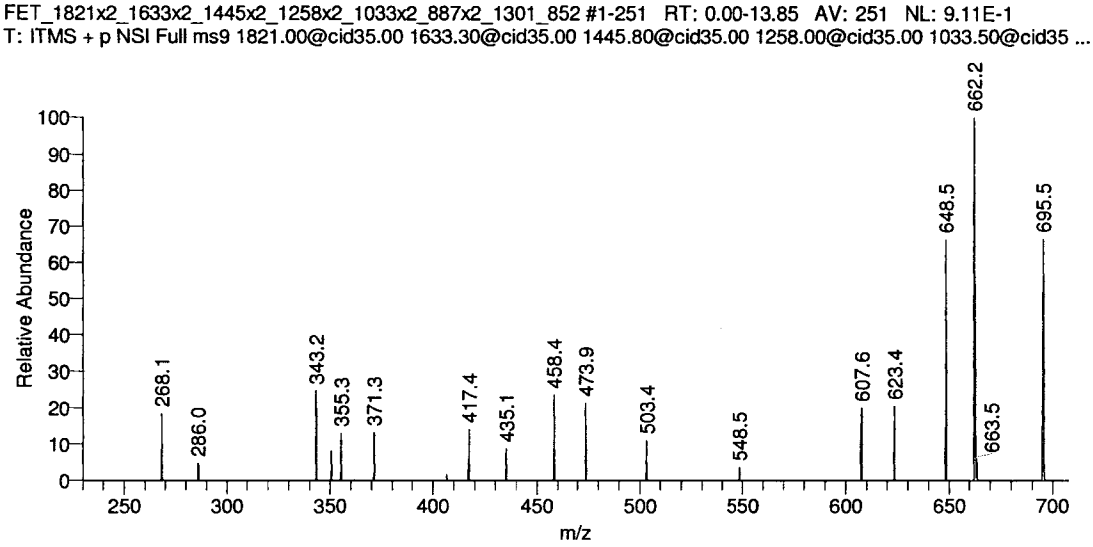
Spectrum A-6: Fetuin *m/z* $1820.9^{2+} \rightarrow 1408.5^{2+} \rightarrow 996.5^{2+}$



Spectrum A-7: Fetuin *m/z* $1820.9^{2+} \rightarrow 847.4$

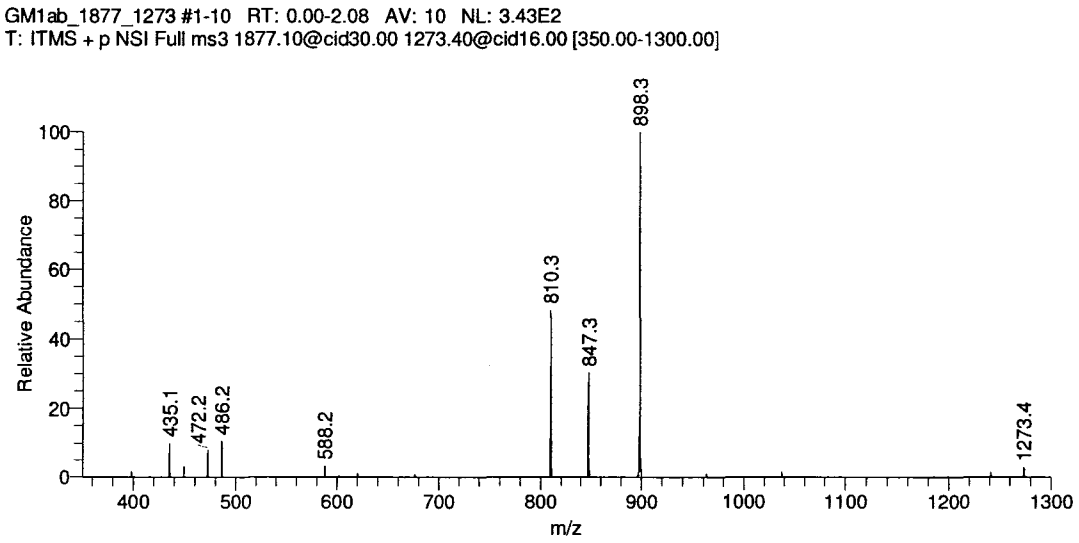


Spectrum A-8: Fetuin *m/z* $1820.9^{2+} \rightarrow 1408.5^{2+} \rightarrow 847.4$

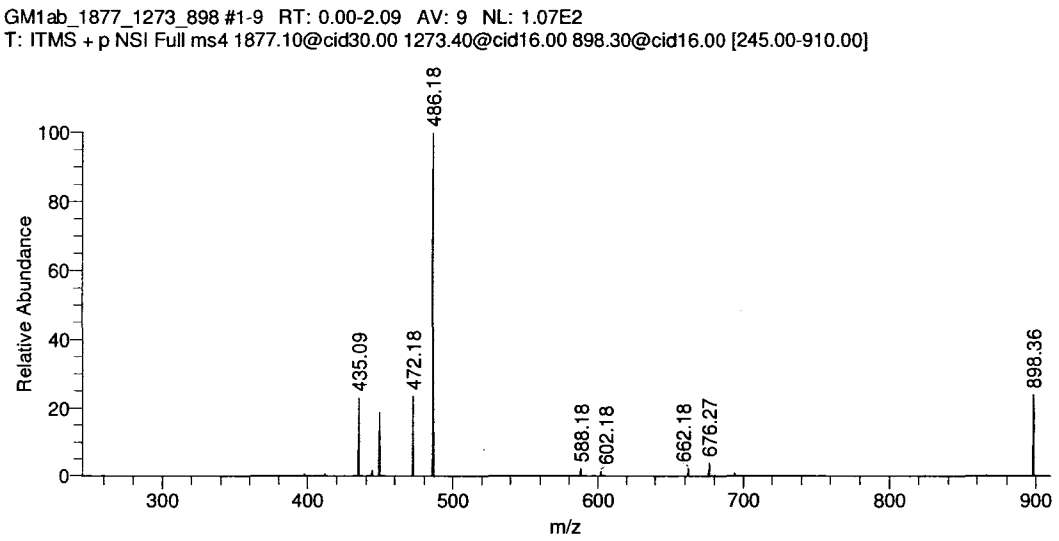


Spectrum A-9: Fetuin *m/z* $1820.2^{2+} \rightarrow 1633.3^{2+} \rightarrow 1445.8^{2+} \rightarrow 1258.0^{2+} \rightarrow 1033.5^{2+} \rightarrow 887.0^{2+} \rightarrow 1301.5 \rightarrow 852.3$

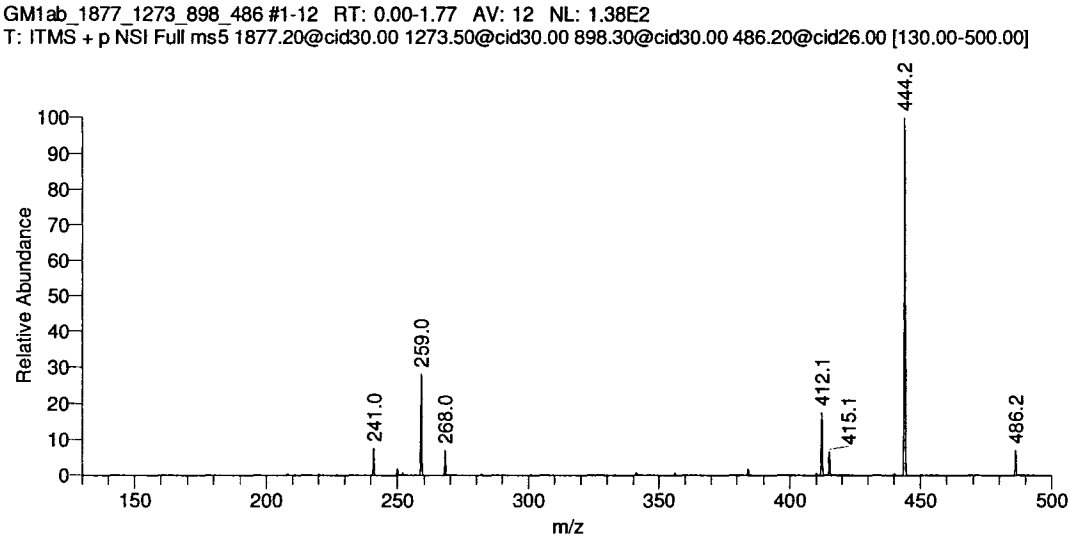
A.2. GM1a/GM1b Spectra



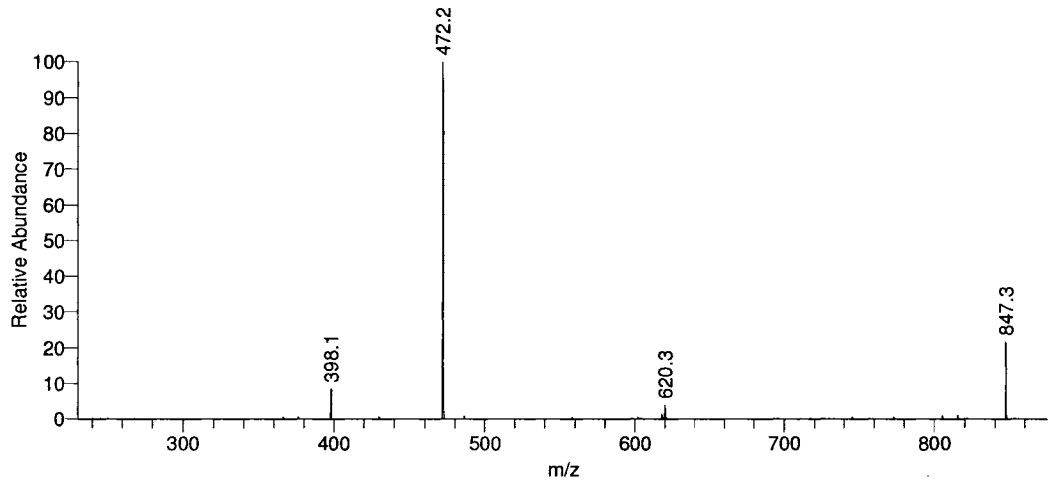
Spectrum A-10: GM1a/GM1b *m/z* 1273.4



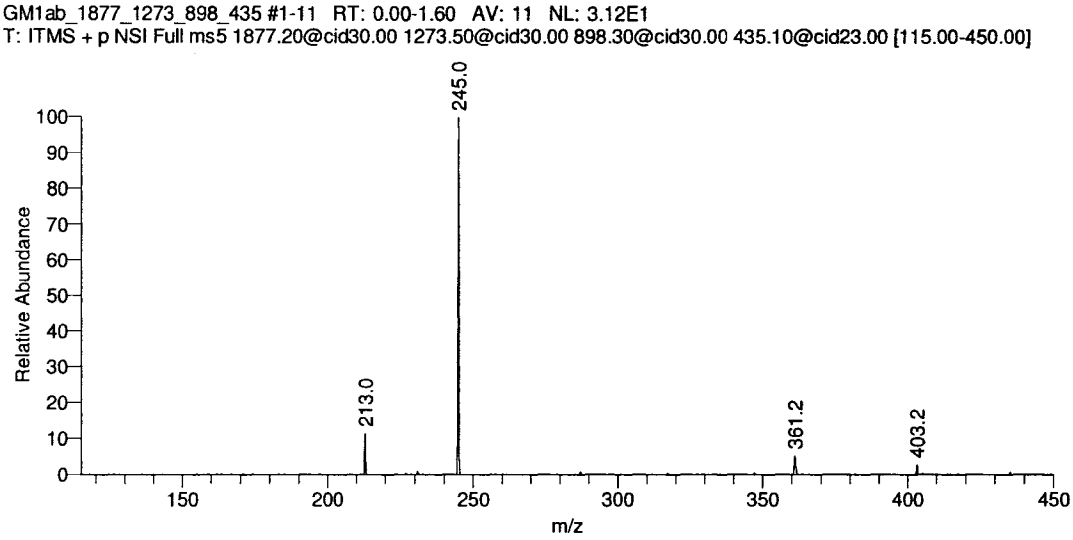
Spectrum A-11: GM1a/GM1b *m/z* 1273.4 → 898.3



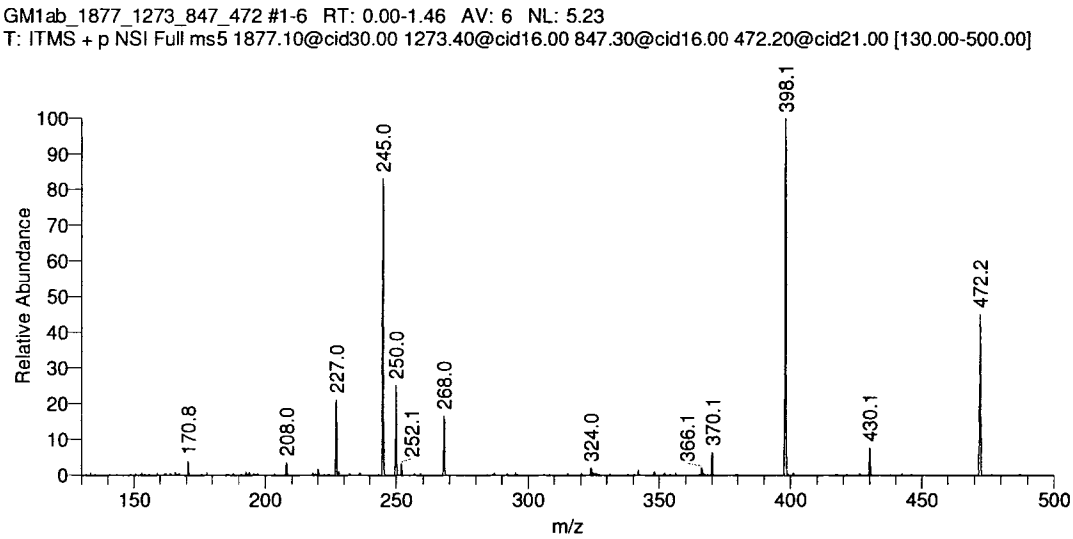
Spectrum A-12: GM1a/GM1b *m/z* 1273.5 → 898.3 → 486.3



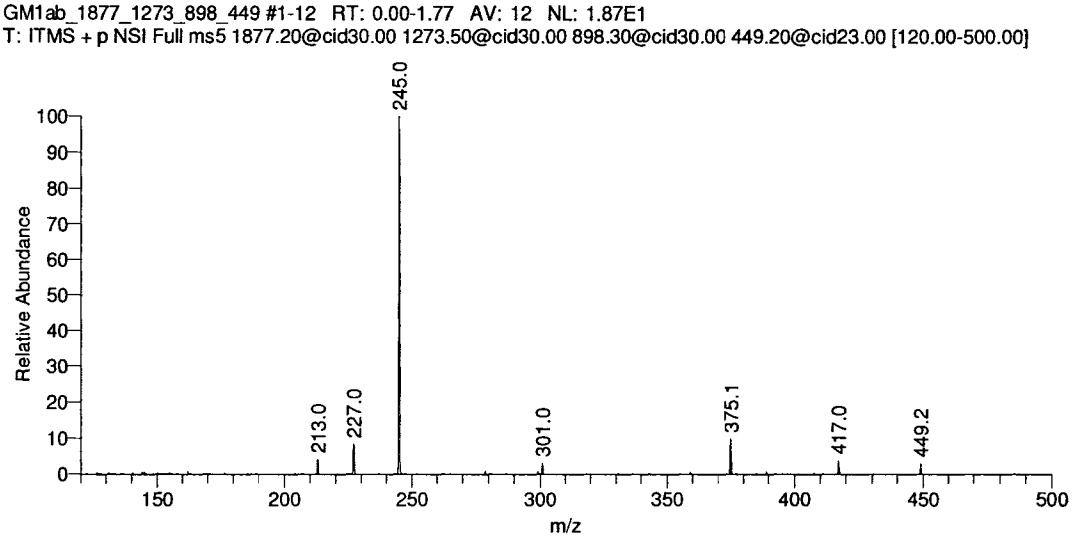
Spectrum A-13: GM1a/GM1b *m/z* 1273.4 → 847.3



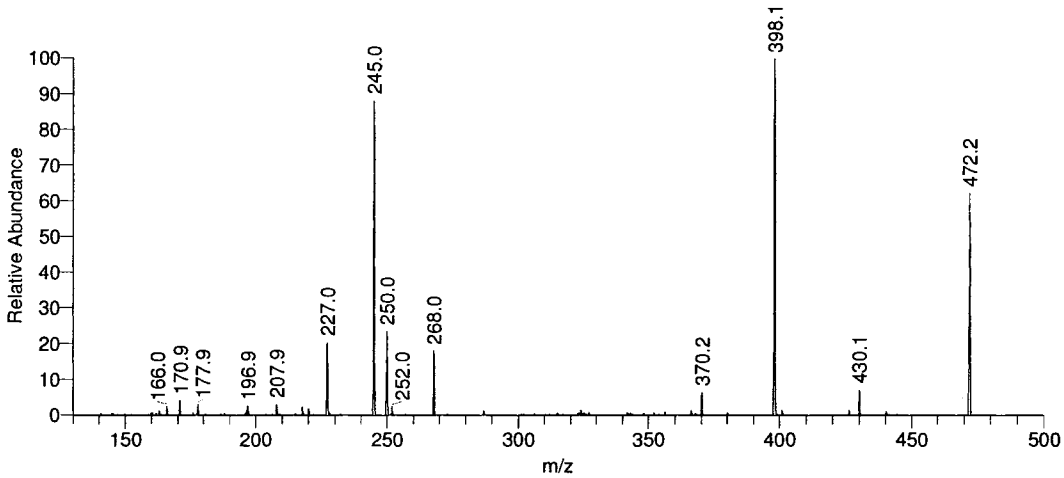
Spectrum A-14: GM1a/GM1b *m/z* 1273.5 → 898.3 → 435.1



Spectrum A-15: GM1a/GM1b *m/z* 1273.4 → 847.3 → 472.2

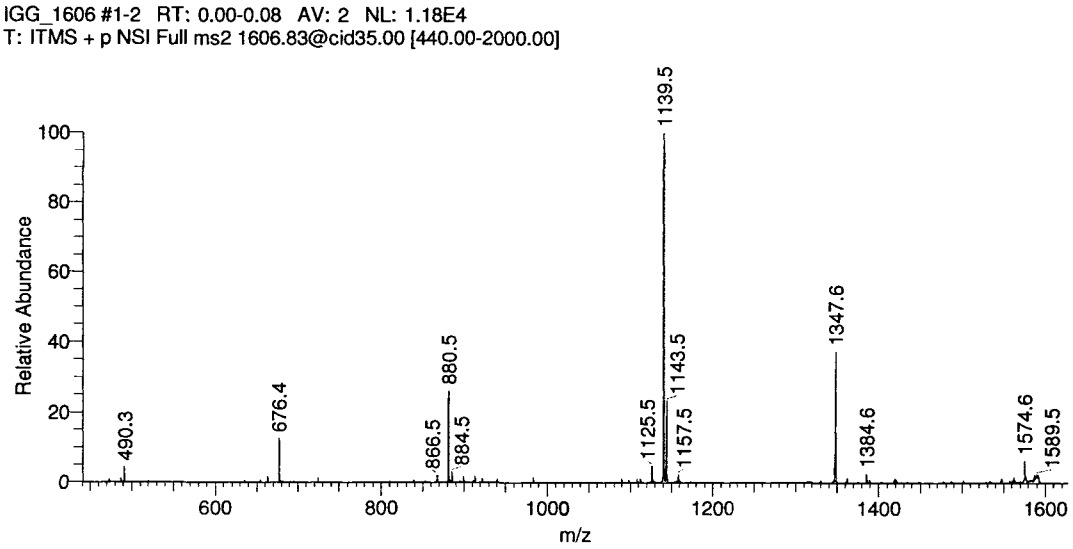


Spectrum A-16: GM1a/GM1b *m/z* 1273.5 → 898.3 → 449.2



Spectrum A-17: GM1a/GM1b *m/z* 1273.5 → 898.3 → 472.2

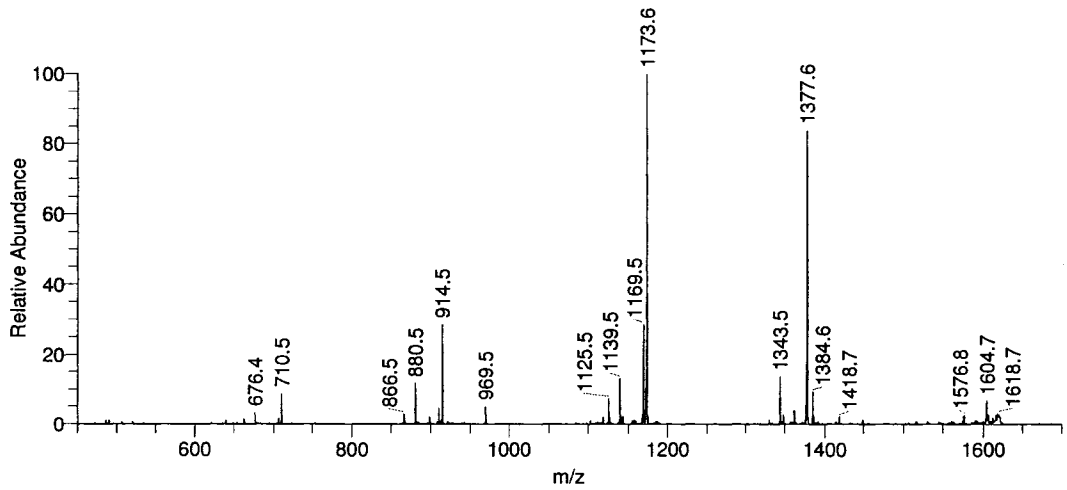
A.3. IgG *m/z* 1606.8 Spectrum



Spectrum A-18: IgG *m/z* 1606.8

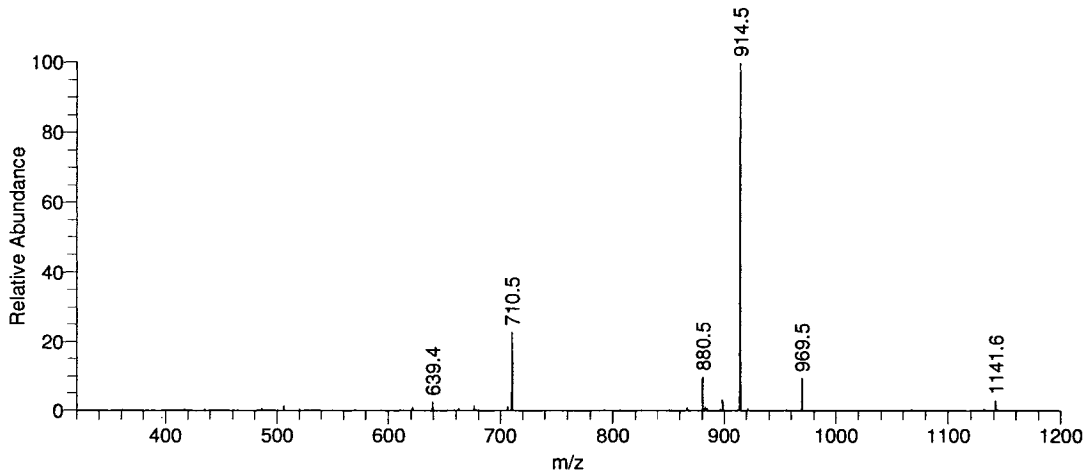
A.4. IgG *m/z* 1636.8 Spectra

IGG_1636 #1-2 RT: 0.00-0.12 AV: 2 NL: 3.02E3
T: ITMS + p NSI Full ms2 1636.84@cid35.00 [450.00-1700.00]

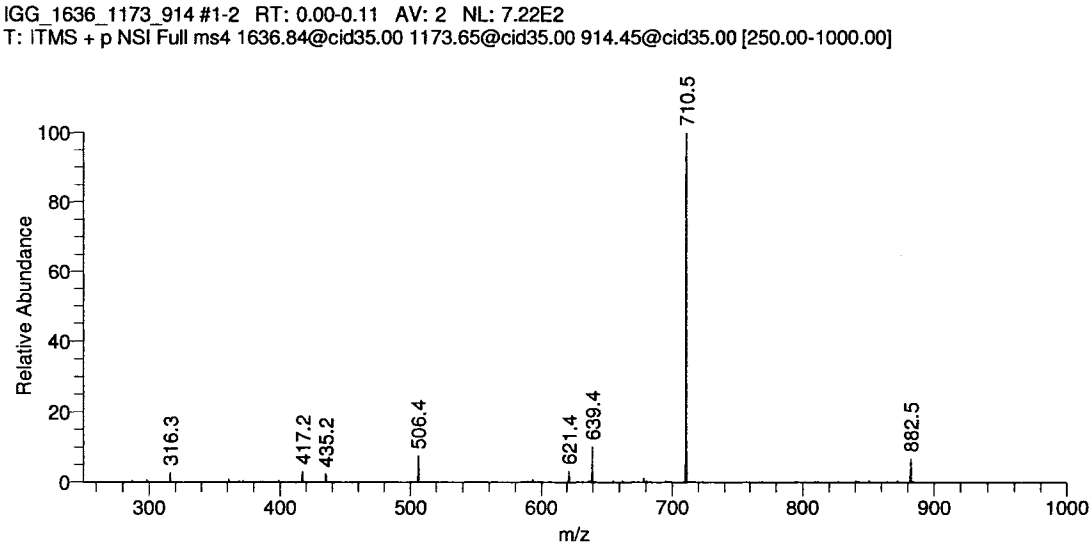


Spectrum A-19: IgG *m/z* 1636.8

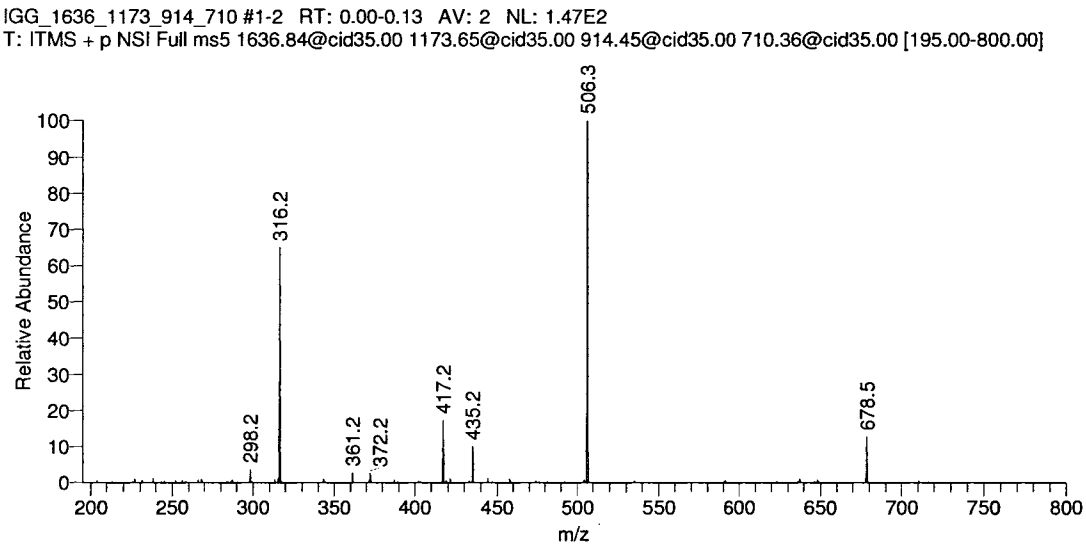
IGG_1636_1173 #1-2 RT: 0.00-0.10 AV: 2 NL: 8.64E2
T: ITMS + p NSI Full ms3 1636.84@cid35.00 1173.65@cid35.00 [320.00-1200.00]



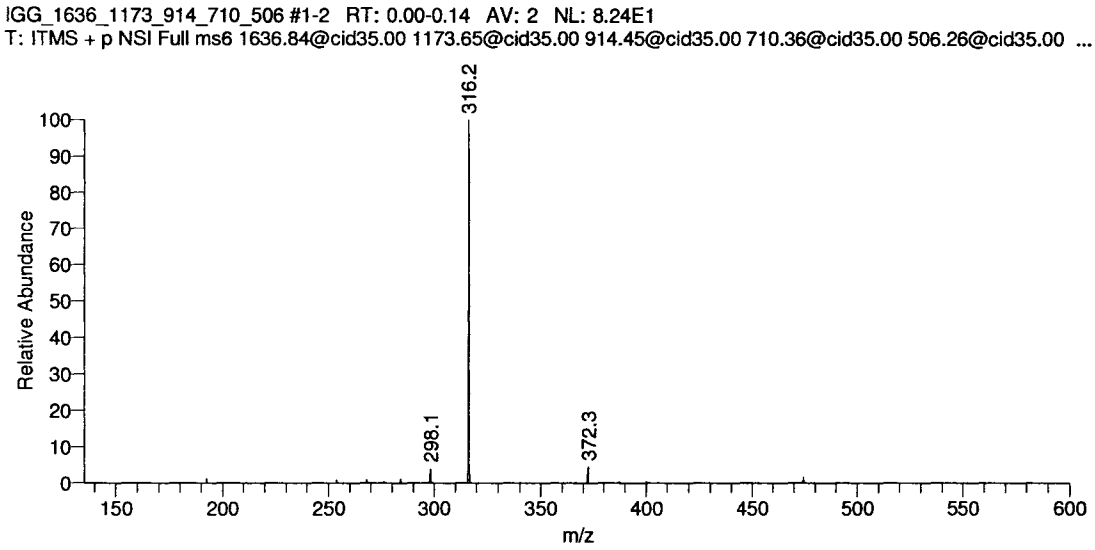
Spectrum A-20: IgG *m/z* 1636.8 → 1173.6



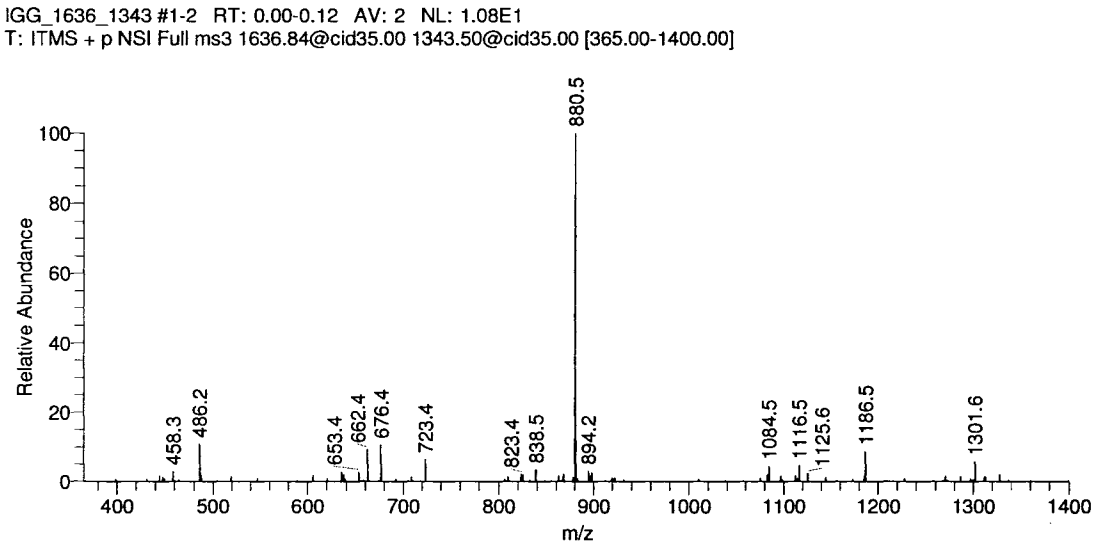
Spectrum A-21: IgG *m/z* 1636.8 → 1173.6 → 914.4



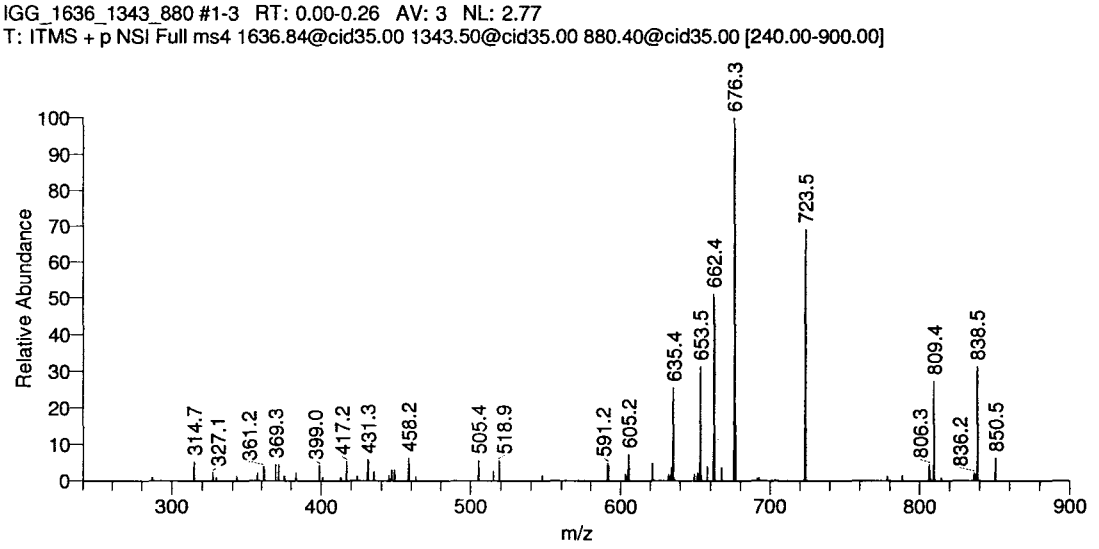
Spectrum A-22: IgG *m/z* 1636.8 → 1173.6 → 914.4 → 710.3



Spectrum A-23: IgG *m/z* 1636.8 → 1173.6 → 914.4 → 710.3 → 506.2



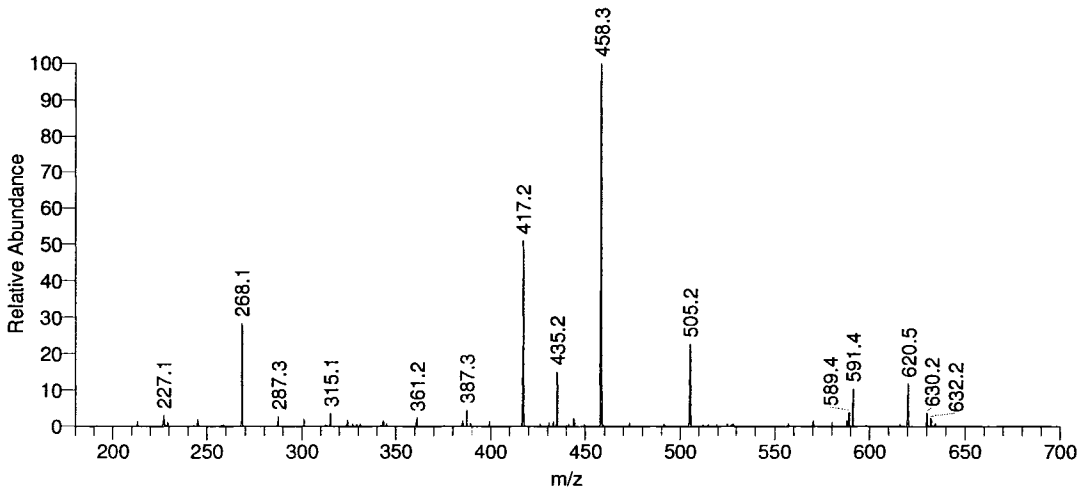
Spectrum A-24: IgG *m/z* 1636.8 → 1343.5



Spectrum A-25: IgG *m/z* 1636.8 → 1343.5 → 880.4

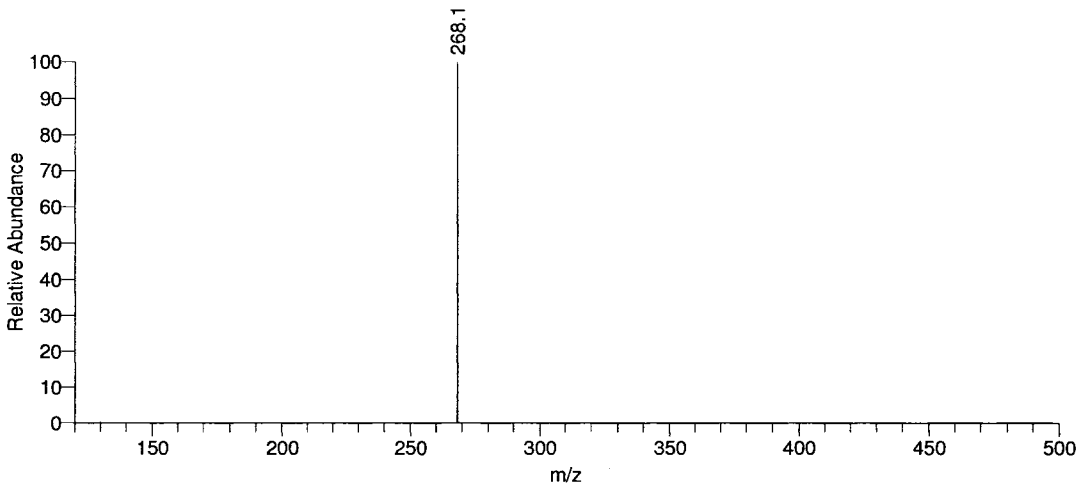
A.5. IgG *m/z* 1677.8 Spectra

IGG_1677_1384_1125_866_662 #1-2 RT: 0.00-0.14 AV: 2 NL: 8.47E1
T: ITMS + p NSI Full ms6 1677.87@cid35.00 1384.64@cid35.00 1125.54@cid35.00 866.40@cid35.00 662.36@cid35.0 ...



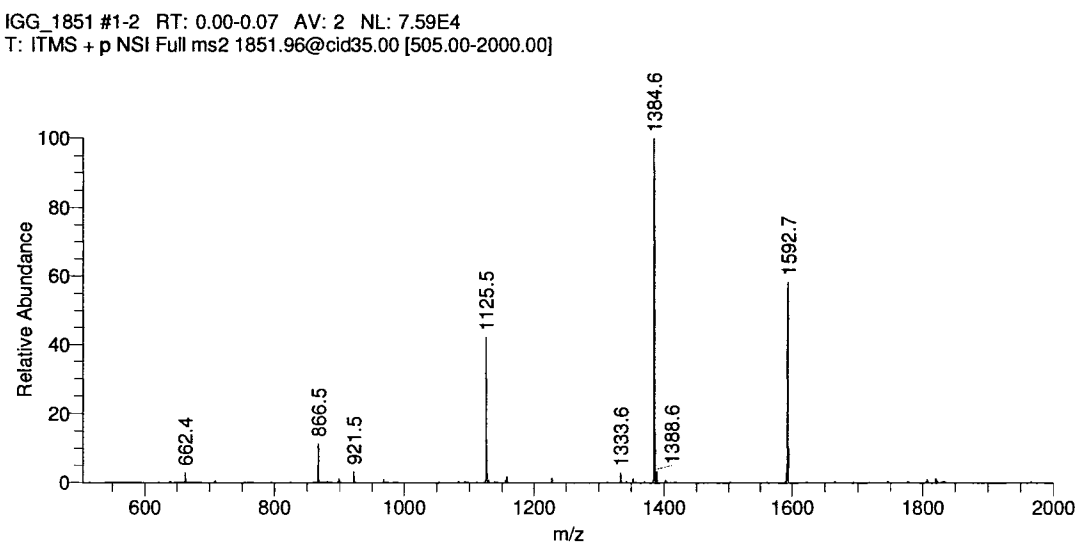
Spectrum A-26: IgG *m/z* 1677.8 → 1384.6 → 1125.5 → 662.4

IGG_1677_1384_1125_866_662_441 #1-20 RT: 0.00-0.92 AV: 20 NL: 3.80E-2
T: ITMS + p NSI Full ms7 1677.80@cid35.00 1384.60@cid35.00 1125.50@cid35.00 866.50@cid35.00 662.40@cid35.0 ...

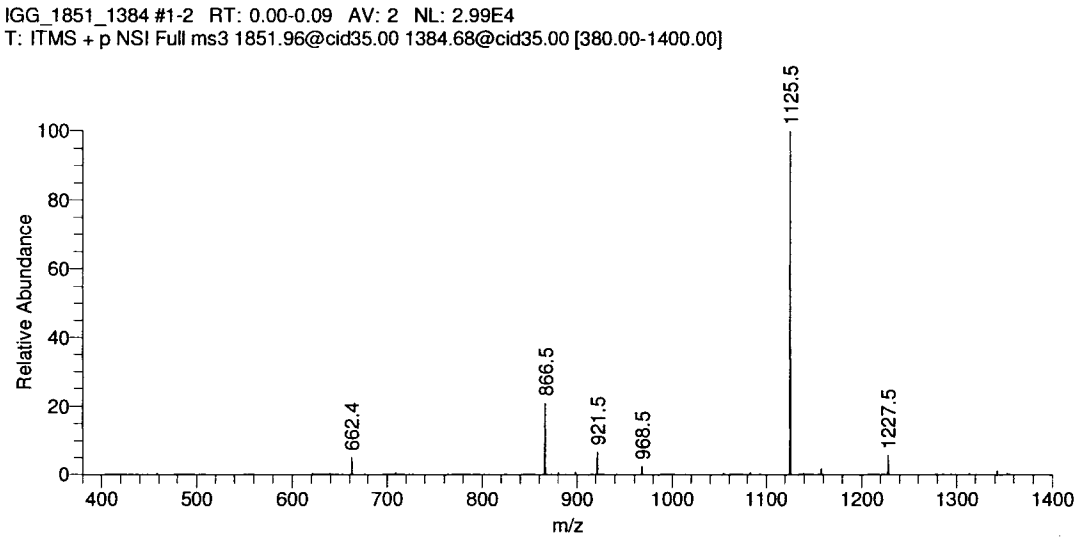


Spectrum A-27: IgG *m/z* 1677.8 → 1384.6 → 1125.5 → 662.4 →441.1

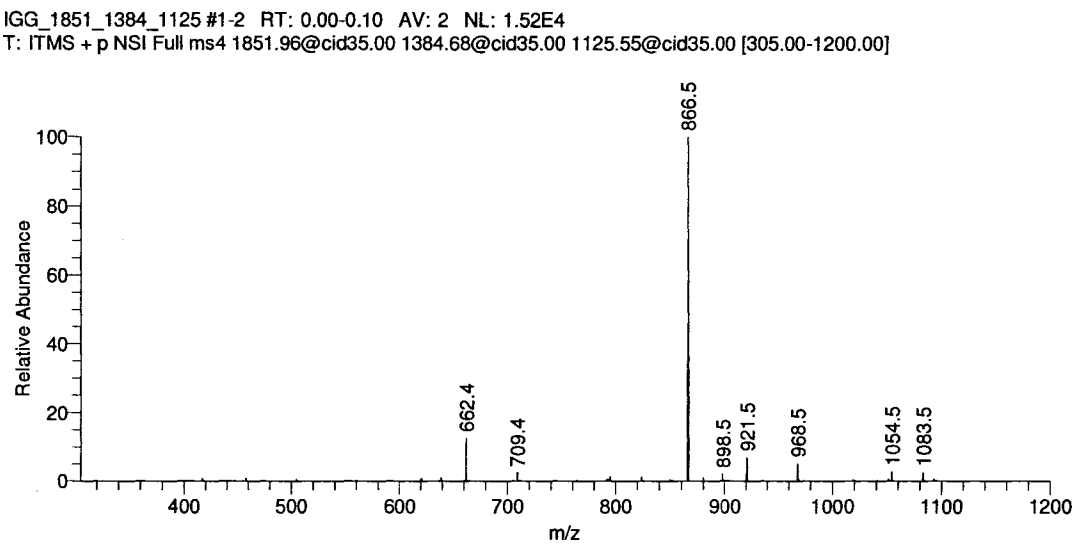
A.6. IgG *m/z* 1851.9 Spectra



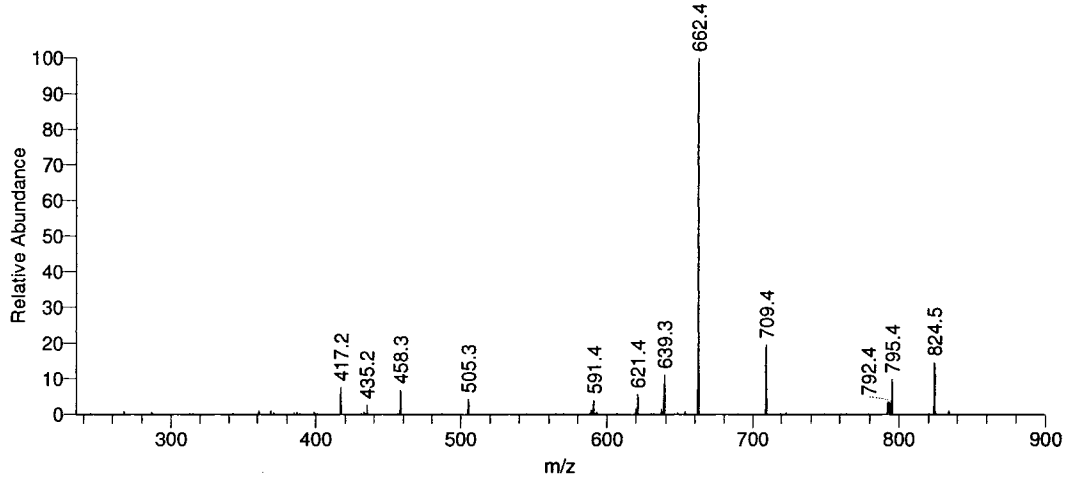
Spectrum A-28: IgG *m/z* 1851.9



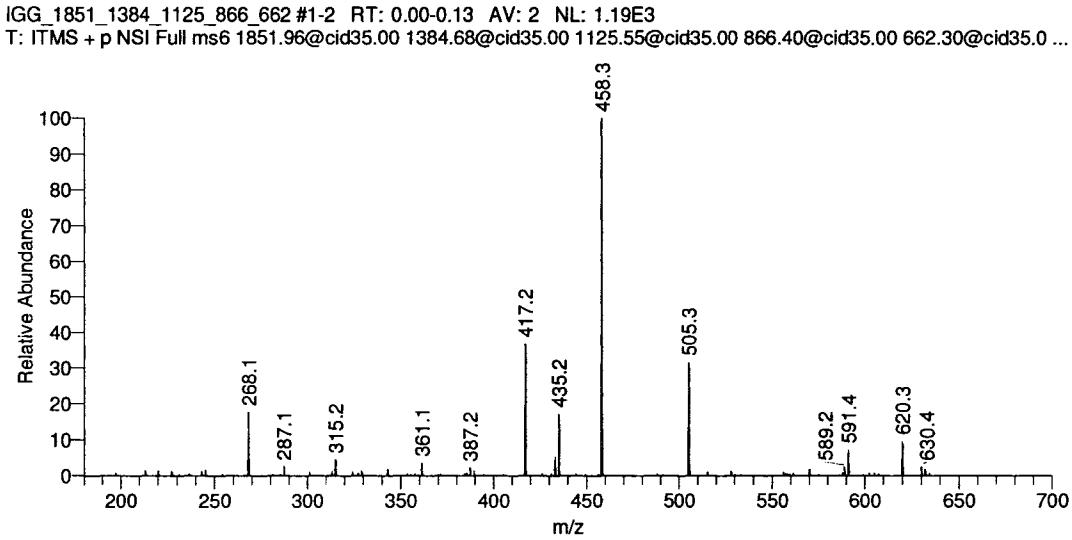
Spectrum A-29: IgG *m/z* 1851.9 → 1384.6



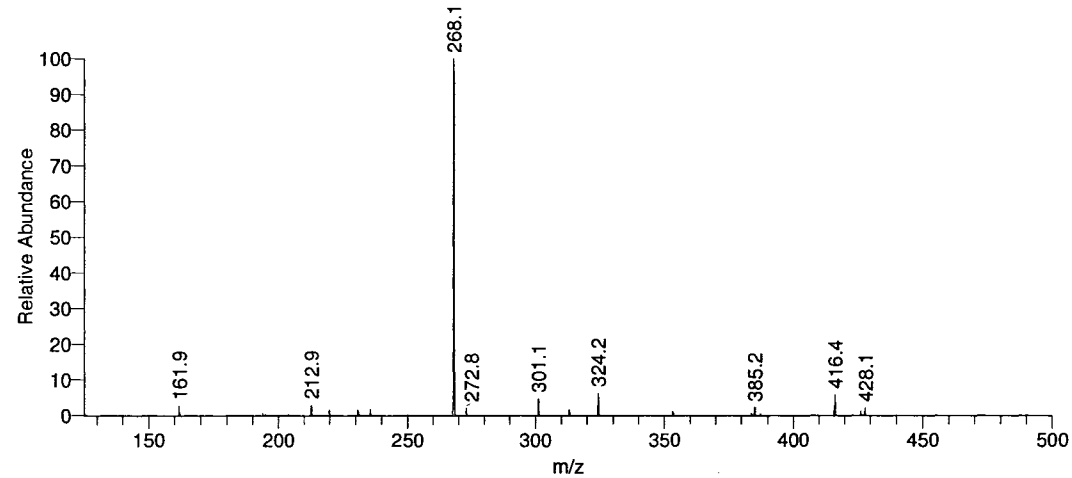
Spectrum A-30: IgG *m/z* 1851.9 → 1384.6 → 1125.5



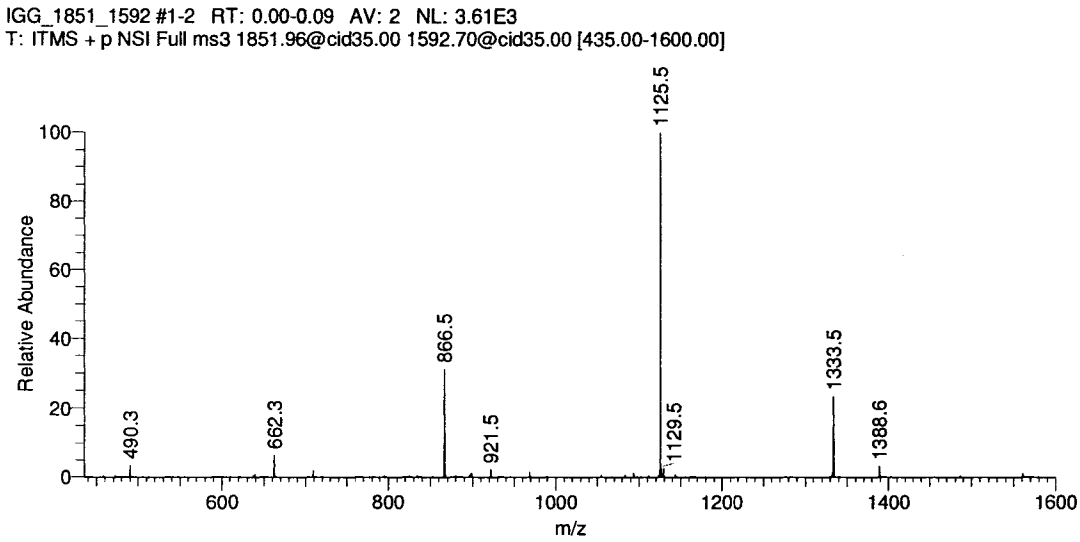
Spectrum A-31: IgG *m/z* 1851.9 → 1384.6 → 1125.5 → 866.4



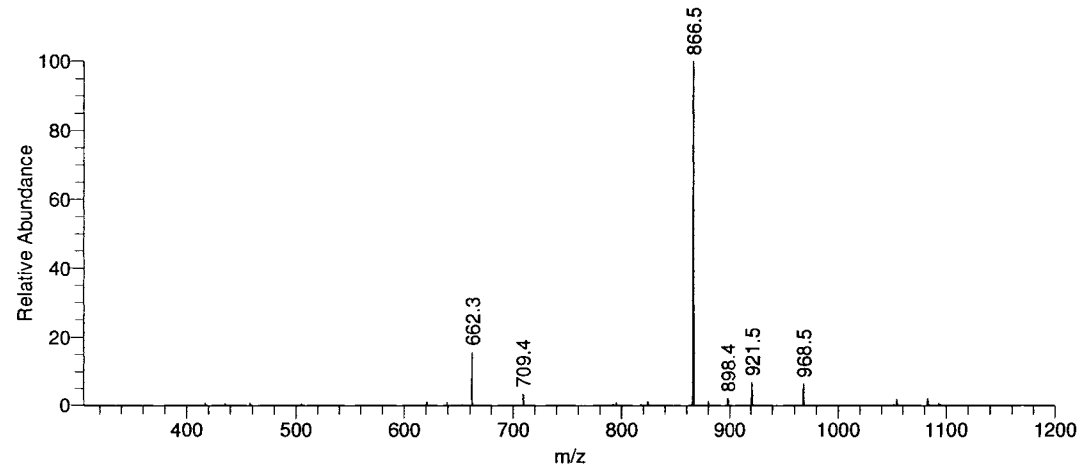
Spectrum A-32: IgG *m/z* 1851.9 → 1384.6 → 1125.5 → 866.4 → 662.3



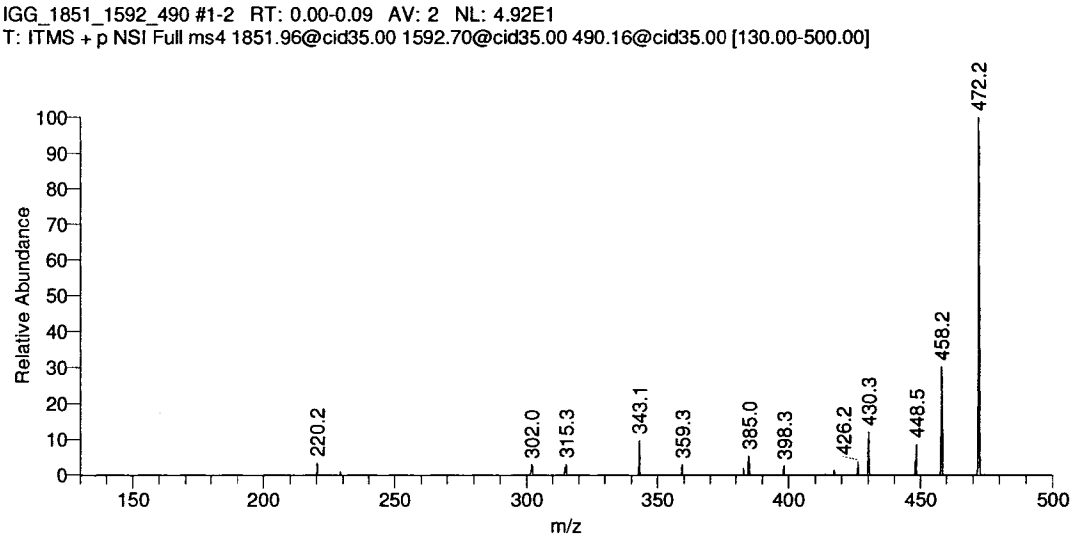
Spectrum A-33: IgG *m/z* 1851.9 → 1384.6 → 1125.5 → 866.4 → 662.3 → 458.2



Spectrum A-34: IgG *m/z* 1851.9 → 1592.7



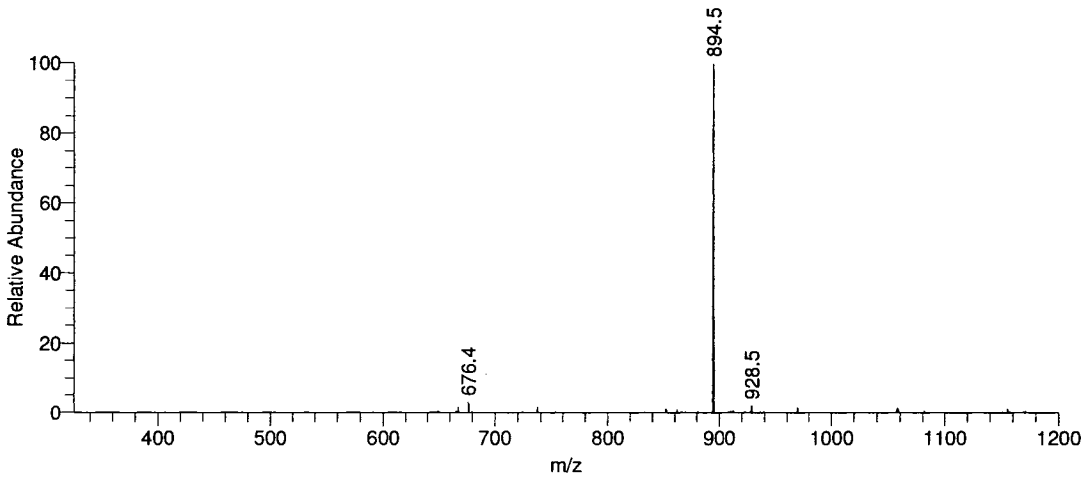
Spectrum A-35: IgG *m/z* 1851.9 → 1592.7 → 1125.4



Spectrum A-36: IgG *m/z* 1851.9 → 1592.7 → 490.1

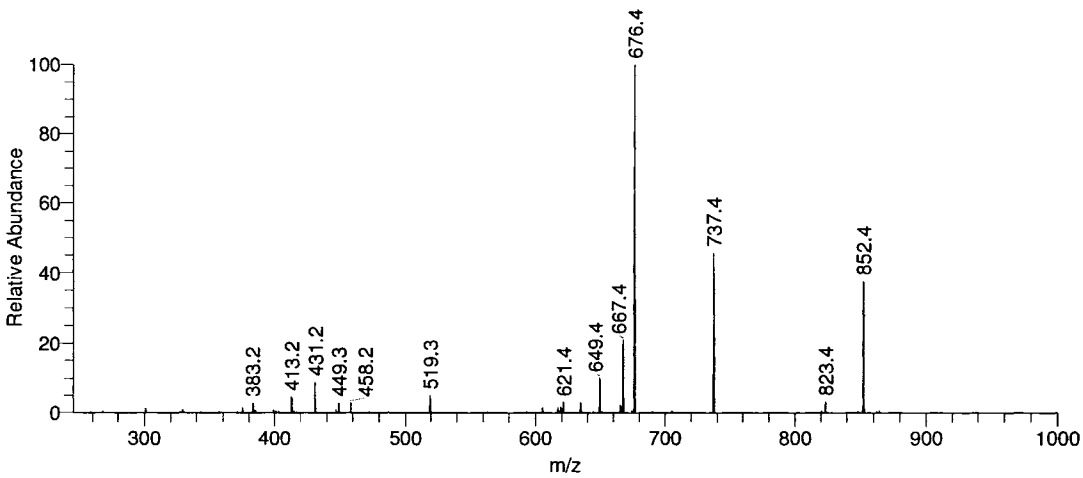
A.7. Ovalbumin *m/z* 1187.6 Spectra

OVA_1187 #1-2 RT: 0.00-0.06 AV: 2 NL: 1.22E5
T: ITMS + p NSI Full ms2 1187.61@cid35.00 [325.00-1200.00]

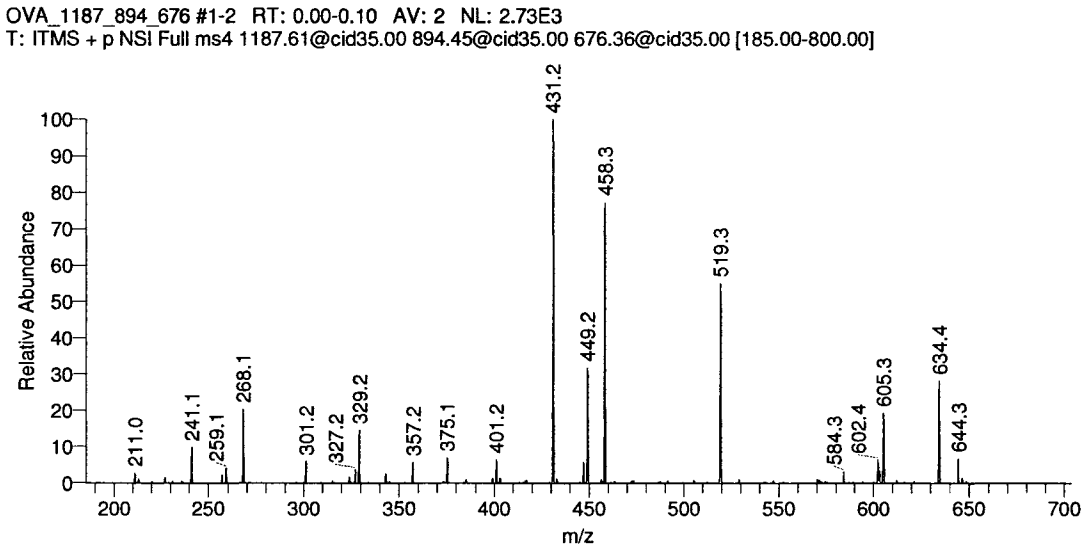


Spectrum A-37: Ovalbumin *m/z* 1187.6

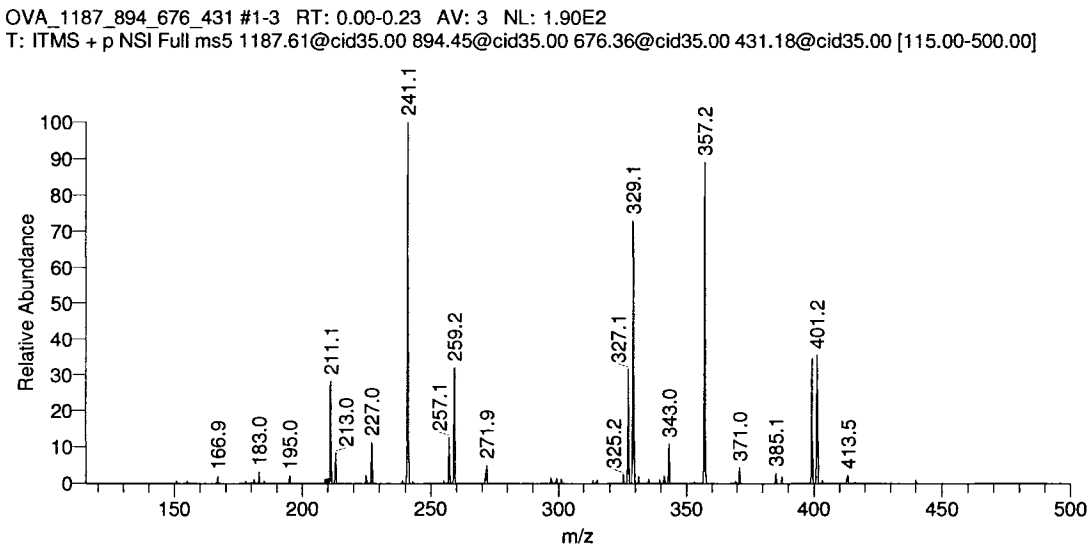
OVA_1187_894 #1-2 RT: 0.00-0.08 AV: 2 NL: 2.84E4
T: ITMS + p NSI Full ms3 1187.61@cid35.00 894.45@cid35.00 [245.00-1000.00]



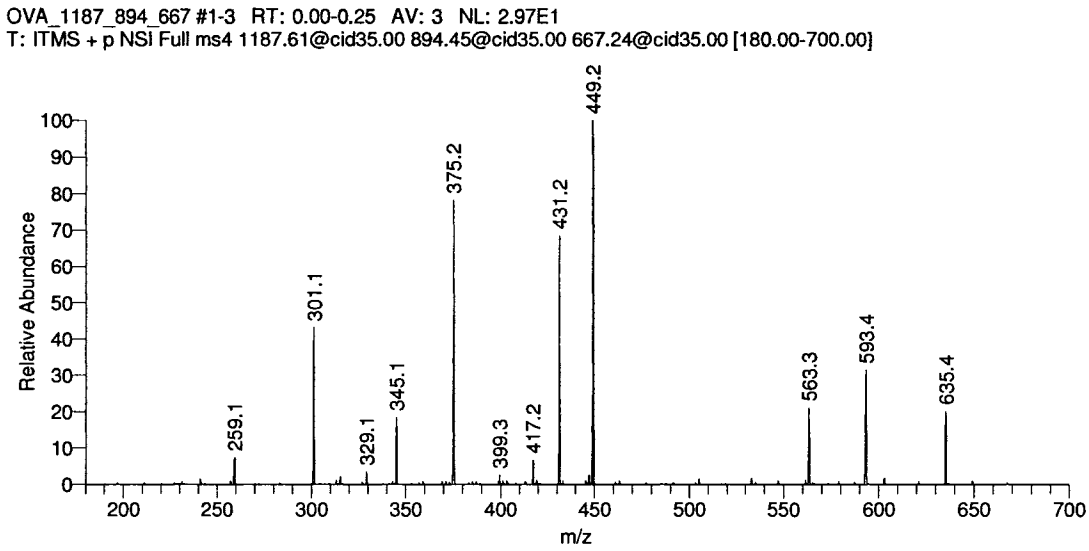
Spectrum A-38: Ovalbumin *m/z* 1187.6 → 898.4



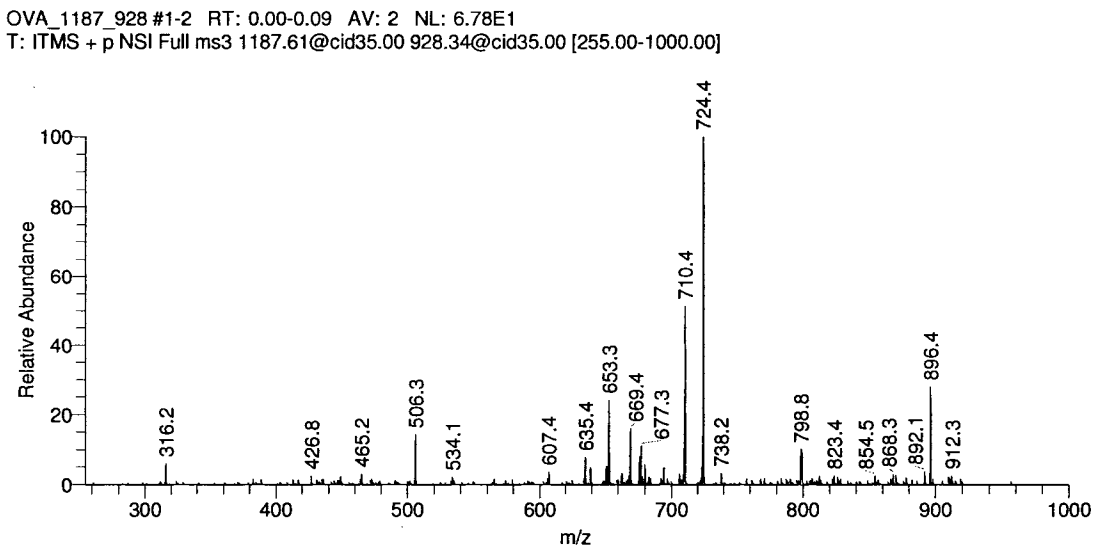
Spectrum A-39: Ovalbumin *m/z* 1187.6 → 898.4 → 676.4



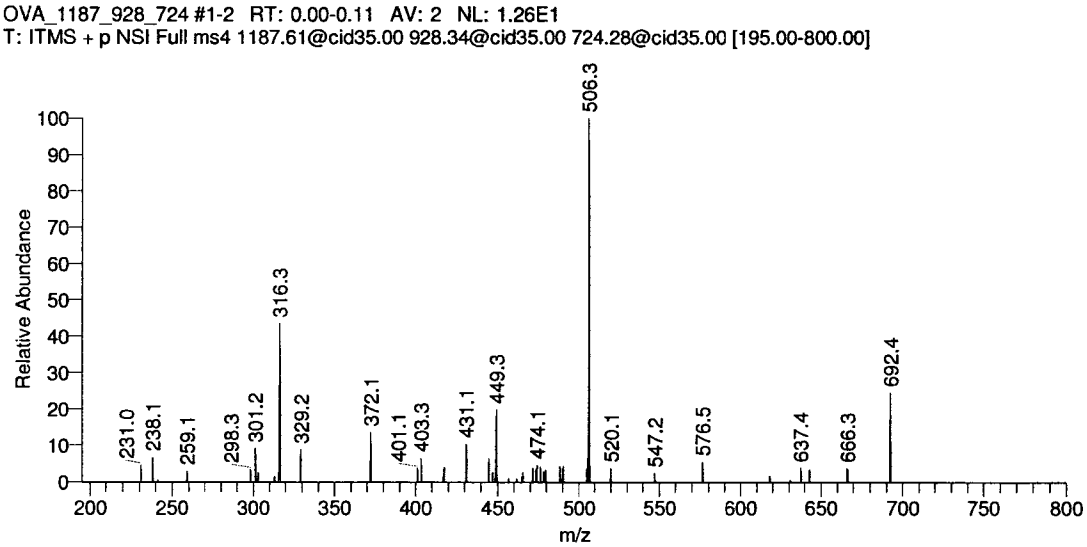
Spectrum A-40: Ovalbumin *m/z* 1187.6 → 898.4 → 676.4 → 431.2



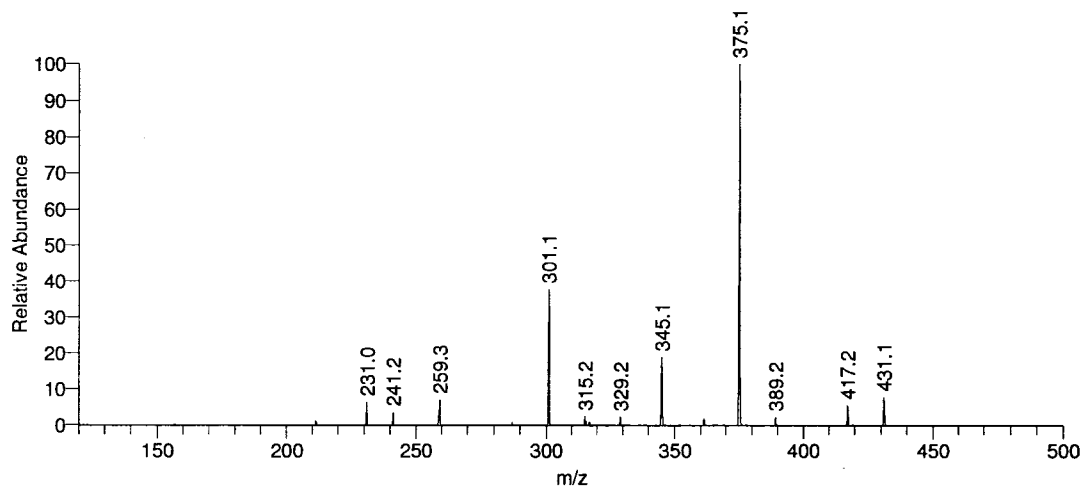
Spectrum A-41: Ovalbumin *m/z* 1187.6 → 898.4 → 667.24



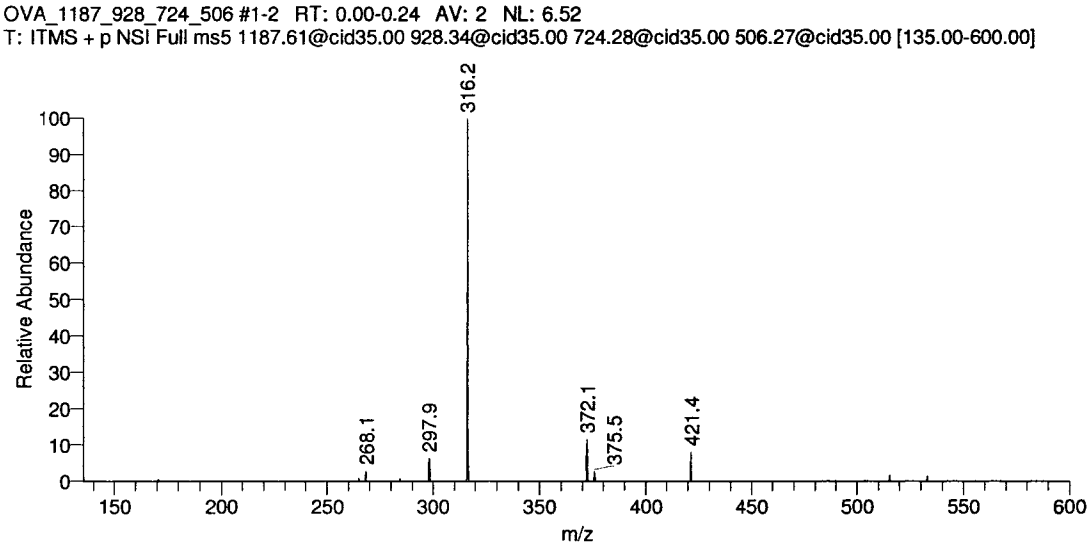
Spectrum A-42: Ovalbumin *m/z* 1187.6 → 928.3



Spectrum A-43: Ovalbumin *m/z* 1187.6 → 928.3 → 724.3

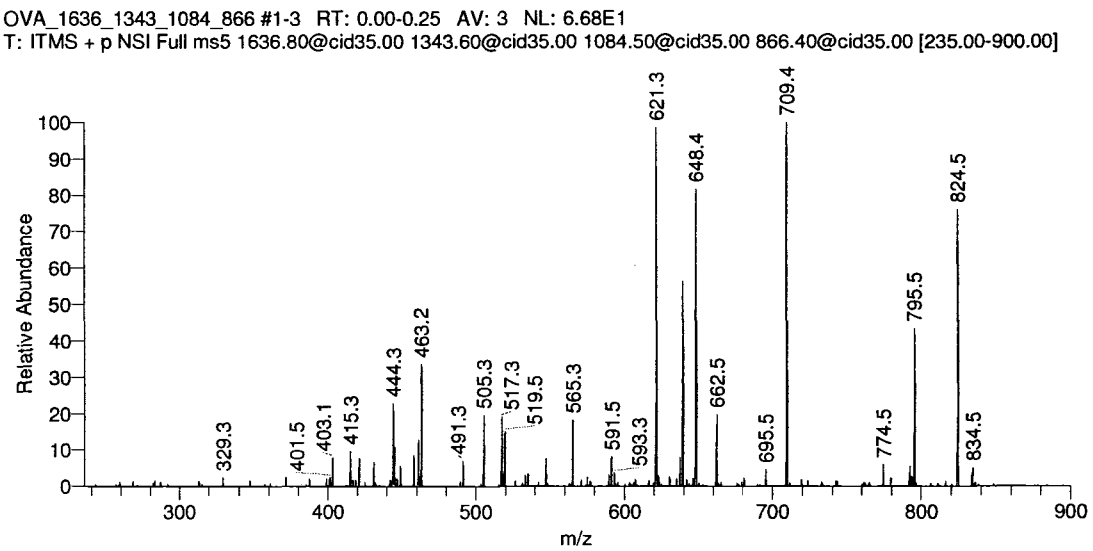


Spectrum A-44: Ovalbumin *m/z* 1187.6 → 898.4 → 667.3 → 449.2

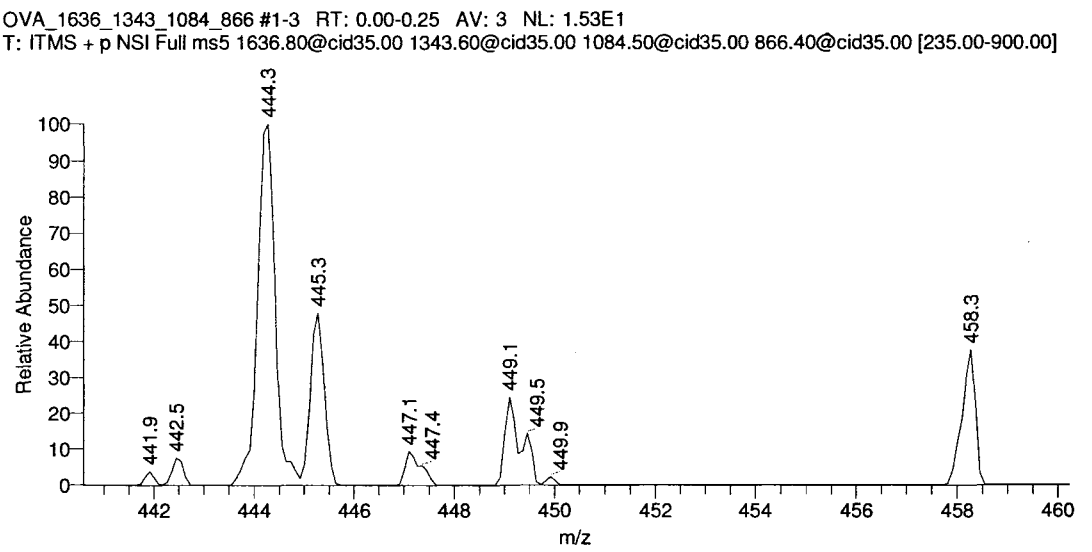


Spectrum A-45: Ovalbumin *m/z* 1187.6 → 928.3 → 724.3 → 506.3

A.8. Ovalbumin *m/z* 1636.8 Spectra



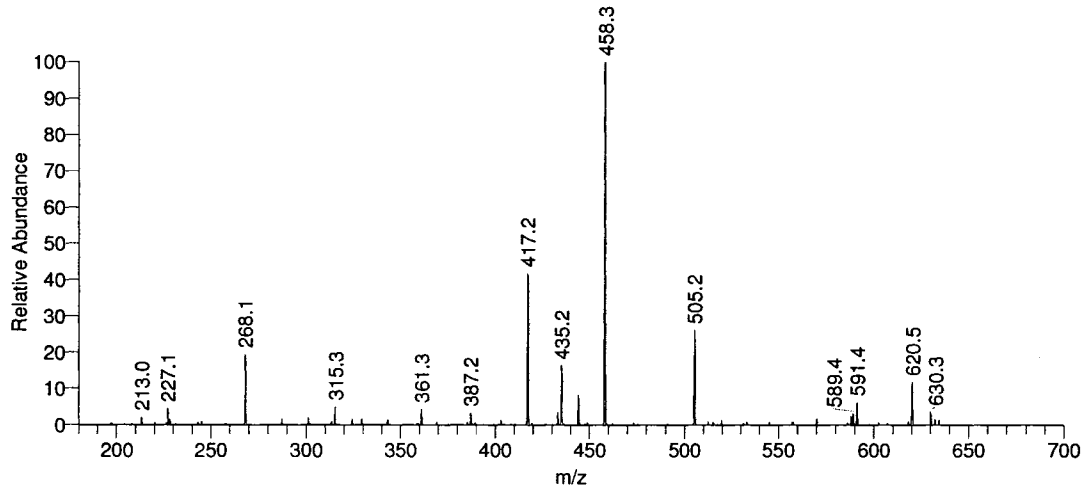
Spectrum A-46: Ovalbumin *m/z* 1636.8 → 1343.6 → 1084.5 → 866.4



Spectrum A-47: Detail from ovalbumin *m/z* 1636.8 → 1343.6 → 1084.5 → 866.4 showing the characteristic ions *m/z* 444 and *m/z* 458 indicating isomers with and without bisecting HexNAcs.

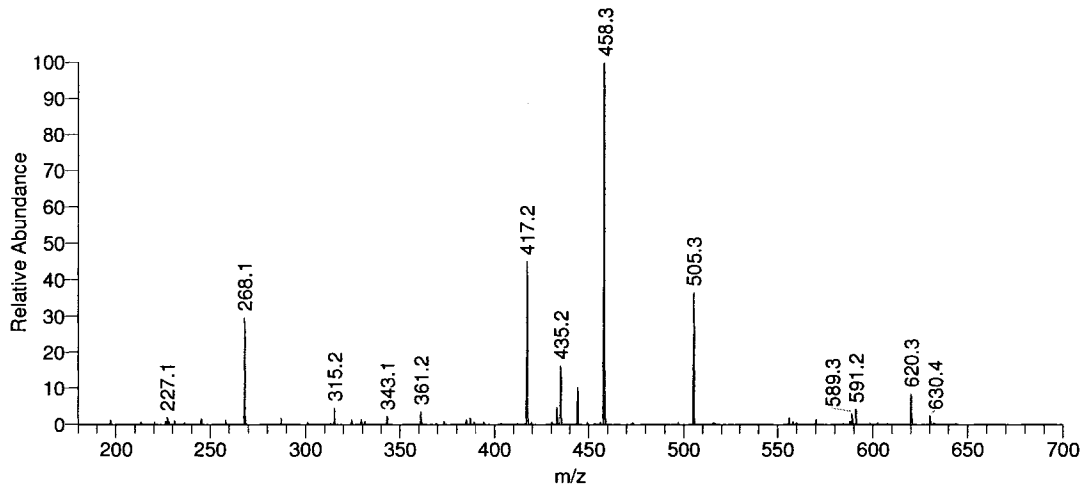
A.9. Ovalbumin *m/z* 1677.8 Spectra

OVA_1677_1384_1125_866_662 #1-3 RT: 0.00-0.27 AV: 3 NL: 3.07E2
T: ITMS + p NSI Full ms6 1677.87@cid35.00 1384.60@cid35.00 1125.50@cid35.00 866.40@cid35.00 662.40@cid35.0 ...

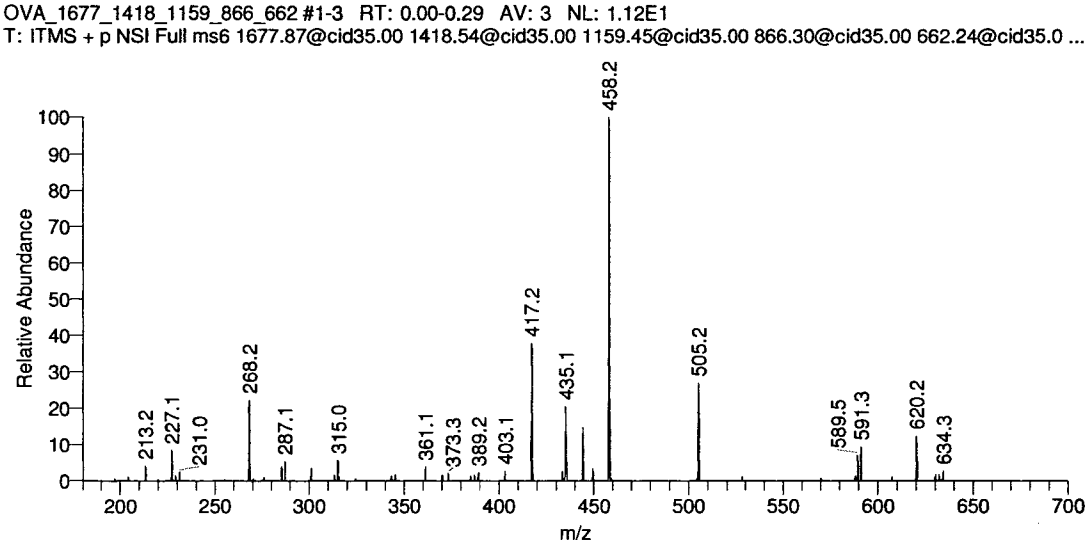


Spectrum A-48: Ovalbumin *m/z* 1677.8 → 1384.6 → 1125.5 → 866.4 → 662.4

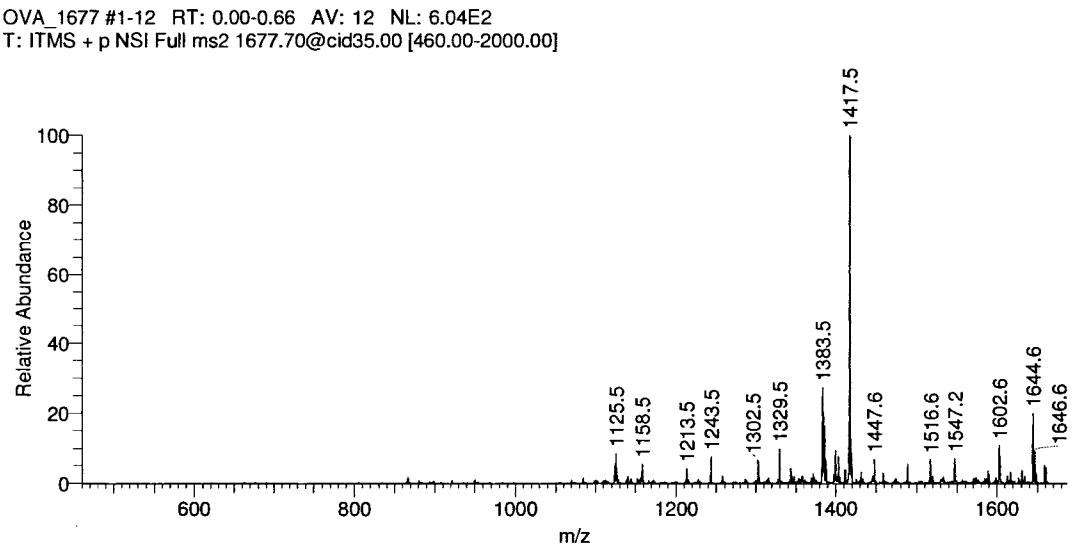
OVA_1677_1418_1125_866_662 #1-3 RT: 0.00-0.30 AV: 3 NL: 3.19E1
T: ITMS + p NSI Full ms6 1677.87@cid35.00 1418.54@cid35.00 1125.39@cid35.00 866.30@cid35.00 662.22@cid35.0 ...



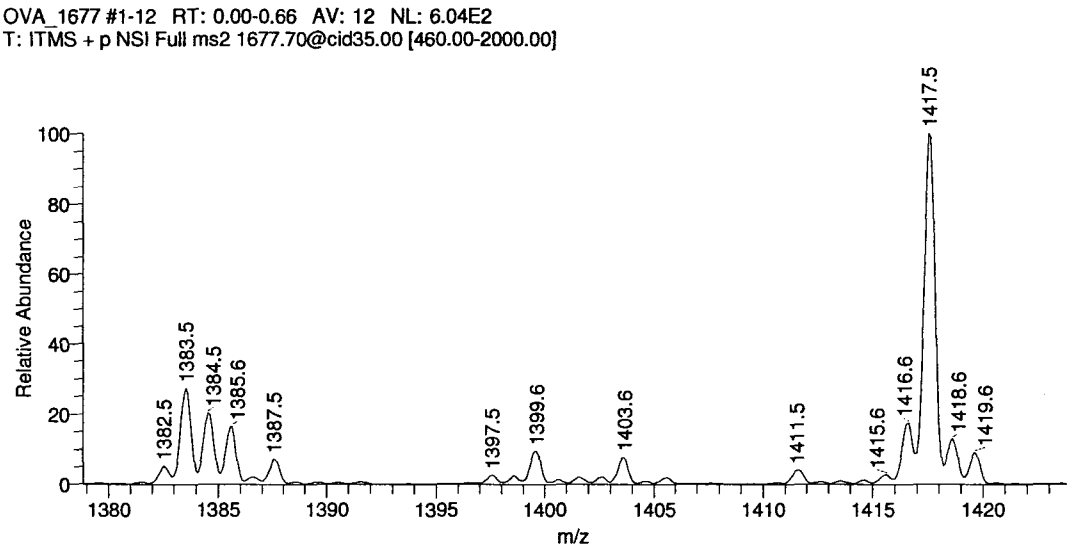
Spectrum A-49: Ovalbumin *m/z* 1677.8 → 1418.5 → 1125.5 → 866.3 → 662.2



Spectrum A-50: Ovalbumin *m/z* 1677.8 → 1418.5 → 1159.4 → 866.3 → 662.3



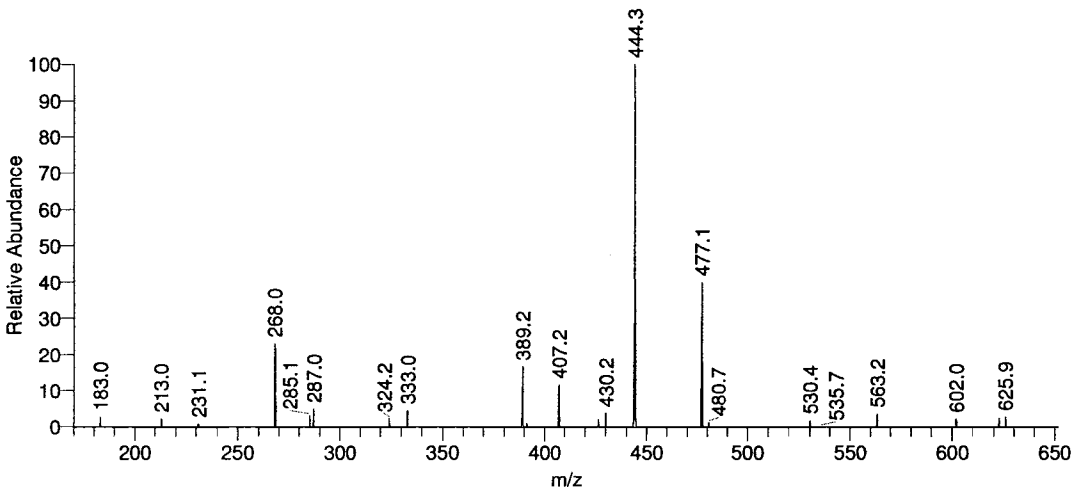
Spectrum A-51: Ovalbumin *m/z* 1677.7



Spectrum A-52: Ovalbumin *m/z* 1677.7 (detail)

A.10. Ovalbumin *m/z* 1923.0 Spectrum

OVA_1923_1663_1370_1111_852_634 #1-2 RT: 0.00-0.20 AV: 2 NL: 2.21
T: ITMS + p NSI Full ms7 1922.99@cid35.00 1663.59@cid35.00 1370.46@cid35.00 1111.40@cid35.00 852.28@cid35. ...



Spectrum A-53: Ovalbumin *m/z* 1923.0 → 1663.6 → 1370.5 → 1111.4 → 852.3 → 634.5

APPENDIX B:

SAMPLE OSCAR INFERENCE RULES

This section describes several of OSCAR's inference rules. The names of the rules are taken directly from the implementing C++ method. The parenthesized phrase following the rule's name identifies the rule's parameter, and indicates whether the rule accepts a mono or a box as its primary argument. The data structure fields used are described in Section 5.3 beginning on page 53.

Note that these rules are described as operating on glycans, boxes, monos, and scars. However, these rules could easily be applied to any domain that includes trees, subtrees, nodes, and dangling edges, respectively. As such, the rules should be considered potentially applicable to a variety of domains. The rare exceptions are the rules that deal with cross-ring cleavages, which are unlikely to map cleanly to other problem spaces.

B.1. InferNumChildrenForSingleton (Box B)

If box B has N child scars and contains a single mono M, then we know that all of those N child scars belong to mono M. We therefore know that M must have exactly N children. We update M's `NumChildrenPossible` field to contain only the value N.

B.2. RootPlusOnlyLeaves (Box B)

If a box B contains N monos and N-1 of those monos are known to be leaves, then the N^{th} mono must be the parent of the N-1 monos. (Since N-1 of the monos cannot have children, only the remaining Nth mono can have children. Furthermore, since all of the monos in the box must form a connected substructure, that N^{th} mono must have all of the other monos as children.) We update the fork by restricting the **ParentPossible** field for each of the N-1 monos to contain only the N^{th} mono.

B.3. ApplyBoxLinkage (Box B)

Suppose that the composition of box B represents a cross-ring fragment that includes only the 6 position of B's parent mono. Clearly, the fragment must have been linked to position 6 of its parent mono and so the only monos that could be the root of box B are those that might be 6-linked to their parent. Therefore, we remove from the box's **RootPossible** field any mono which does not have 6 in its **LinkageMonoToParentPossible** field.

B.4. RestrictParentPossibleGivenCrossRingBox (Box B)

Given a box B that has a cross-ring cleavage composition and **RootDefinite** RD, we can restrict RD's possible parents to those that have enough possible children to account for the child scars on the cross-ring fragment.

For example, suppose box B has a ^{3,5}A cross-ring cleavage that contains one glycosidic bond and one (oh) scar. RD's parent must therefore have at least two children: linkage positions 4 and 6 are both known to be occupied. Any possible parent of RD has less than two children (**NumChildrenPossible**), or that does not have both position 4 and 6 available to attach children (**LinkageMonoToChildrenPossible**), is eliminated from box B's **RootParent-Possible** field.

B.5. ApplyLeaf (Mono M)

If we know that the mono M is a leaf (because **NumChildrenPossible** = { 0 }), then the inference rule performs the following updates:

1) Clear mono M's **ChildrenPossible** to empty (because the mono has no children).

2) Remove mono M from the **ParentPossible** field of all other monos (because mono M cannot be the parent of any mono).

3) Remove mono M from the **RootParentPossible** field of all boxes in the enclosing fork (because no fragment can attach to this mono).

4) Remove mono M from the **RootPossible** field of all boxes which contain more than one mono (since some other mono in those boxes must be the roots).

B.6. NoPossibleParentsImpliesMSRoot (Mono M)

If the mono M has no possible parents (**ParentPossible** is empty), then the mono must be the root of the glycan. The containing fork is annotated appropriately.

B.7. ApplyMSRootToAnnMono (Mono M)

If mono M is known to be the root of the glycan (for example, as the result of the previous inference rule, NoPossibleParentsImpliesMSRoot), this inference rule:

1) Clears M's **ParentPossible** field (because the root cannot have a parent) and

2) Removes M from the **ChildrenPossible** field of all monos in the fork (because the root cannot be a child).

B.8. ApplyMSRootToAnnBox (Box B)

If box B contains mono M, where M is known to be the root of the glycan, then M must also be the root of the box. Restrict box B's `RootPossible` field by excluding all monos except M.

B.9. AllChildrenAccountedFor (Mono M)

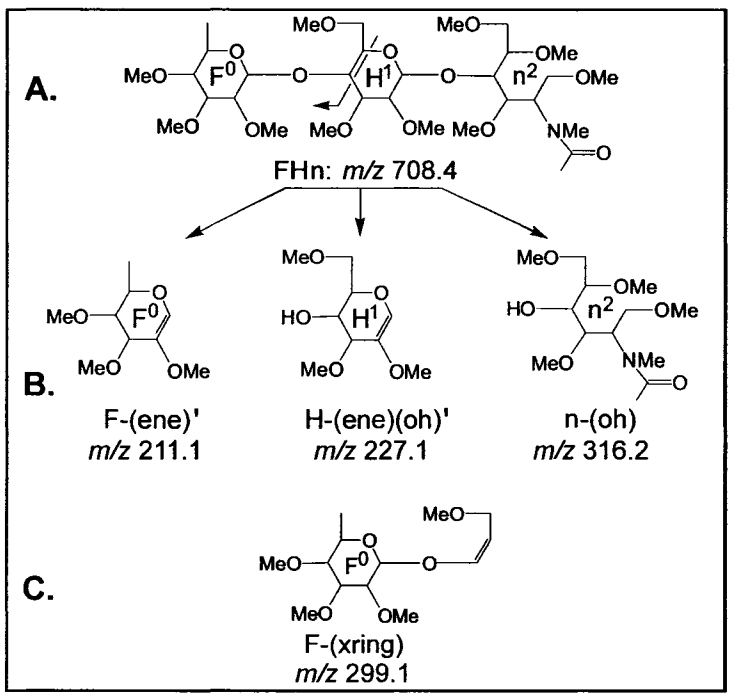
Suppose mono M has exactly N children (`NumChildrenPossible` = { N }) and those children are all known (`ChildrenDefinite` contains exactly N monos). We therefore know that all children of mono M have been found, and update the enclosing fork by removing M from the `ParentPossible` field of all monos other than M's definite children.

B.10. AssignChildLinkage (Mono M)

If parent mono M has a definite child mono C (that is, `M.ChildrenDefinite` contains C) and C has a definite linkage L to its parent mono (`C.Linkage` contains the single value L), then we know that linkage position L on parent mono M is occupied by child C. Obviously, no other child of M can share this linkage position, and so the inference rule updates all other definite children of M by removing L from their `Linkage` field.

B.11. InferNumChildrenFromCrossRingCleavage (Box B)

Referring to the following illustration, assume that OSCAR has deduced that the cross-ring fragment in (C) must have come from the mono H^1. Because the cross-ring fragment contains two linkage positions (4 and 6), and one is unscarred (in this case, position 6), we can infer that H^1 has at most three children. (H^1 could have four children only if positions 2, 3, 4, and 6 were all occupied, but we know that 6 is not scarred.) The inference rule updates the H^1 mono by removing 4 from its **`NumChildrenPossible`** field. This is an example of a rule that requires chemical knowledge—specifically, the structure of cross-ring fragments—in order to derive details of the original glycan.



REFERENCES CITED

(1) Ada, G.; Isaacs, D. *Clin Microbiol Infect.* "Carbohydrate-protein conjugate vaccines", 2003, *9* (2), 79-85.

(2) Alper, J. In *Science*, "Turning Sweet on Cancer", 2003; Vol. 301.

(3) Aoki, K. F.; Yamaguchi, A.; Ueda, N.; Akutsu, T.; Mamitsuka, H.; Goto, S.; Kanehisa, M. *Nucleic Acids Research* "KCaM (KEGG Carbohydrate Matcher): a software tool for analyzing the structures of carbohydrate sugar chains", 2004, *32* (Web Server Issue), W267–W272.

(4) Apweiler, R.; Hermjakob, H.; Sharon, N. *Biochim Biophys Acta.* "On the frequency of protein glycosylation, as deduced from analysis of the SWISS-PROT database", 1999, *1473* (1), 4-8.

(5) Ashline, D.; Singh, S.; Hanneman, A.; Reinhold, V. *Anal. Chem.* "Congruent Strategies for Carbohydrate Sequencing: 1. Mining Structural Details by MS^n", 2005, *77* (19), 6250-6262.

(6) Ashline, D. J.; Lapadula, A. J.; Liu, Y.-H.; Lin, M.; Grace, M.; Pramanik, B.; Reinhold, V. N. *Anal. Chem.* "Carbohydrate Structural Isomers Analyzed by Sequential Mass Spectrometry", 2007, *79* (10), 3830-3842.

(7) Ashline, D. J.; Lapadula, A. J.; Reinhold, V., 54th ASMS Conference on Mass Spectrometry, "Analysis of Isobaric Oligosaccharide Mixtures by Sequential Mass Spectrometry (Poster ThP 302)", Seattle, WA, May 28 - June 1, 2006.

(8) Ashline, D. J.; Lapadula, A. J.; Reinhold, V. N., 53rd ASMS Conference, "Automated Data Collection for Sequential Mass Spectrometry of Glycans." San Antonio, Texas, USA, June 5-9, 2005.

(9) Ashline, D. J.; Lapadula, A. J.; Reinhold, V. N. "Isomeric N-linked Oligosaccharides in IgG Containing Reducing-end Hexose and Reducing-end Fucose Determined by Sequential Mass Spectrometry", 2007 (manuscript in preparation).

(10) Brooks, S. A.; Dwek, M. V.; Schumacher, U. *Functional and Molecular Glycobiology*; BIOS Scientific Publishers Limited: Oxford, UK, 2002.

(11) Brown, W. H. *Introduction to Organic Chemistry*; Saunders College Publishing, 1997.

(12) Buchanan, B. G.; Sutherland, G. L.; Feigenbaum, E. A. In *Machine Intelligence 4*; Meltzer, B., Michie, D., Swann, M., Eds.; Edinburgh University Press: Edinburgh, Scotland, 1969, pp 209-254.

(13) Butler, M.; Quelhas, D.; Critchley, A. J.; Carchon, H.; Hebestreit, H. F.; Hibbert, R. G.; Vilarinho, L.; Teles, E.; Matthijs, G.; Schollen, E.; Argibay, P.; Harvey, D. J.; Dwek, R.

A.; Jaeken, J.; Rudd, P. M. *Glycobiology* "Detailed glycan analysis of serum glycoproteins of patients with congenital disorders of glycosylation indicates the specific defective glycan processing step and provides an insight into pathogenesis", 2003, *13* (9), 601-622.

(14) Butters, T. D.; Dwek, R. A.; Platt, F. M. *Adv Exp Med Biol.* "New therapeutics for the treatment of glycosphingolipid lysosomal storage diseases", 2003, *535*, 219-226.

(15) Campbell, M. K.; Farrell, S. O. *Biochemistry*, 4 ed.; Thomson Brooks/Cole, 2003.

(16) Cancilla, M. T.; Penn, S. G.; Lebrilla, C. B. *Anal. Chem.* "Alkaline Degradation of Oligosaccharides Coupled with Matrix-Assisted Laser Desorption/Ionization Fourier Transform Mass Spectrometry: A Method for Sequencing Oligosaccharides", 1998, *70*, 663-672.

(17) Ciucanu, I.; Kerek, F. *Carbohydr. Res.* "A simple and rapid method for the permethylation of carbohydrates", 1984, *131*, 209-217.

(18) Cooper, C. A.; Gasteiger, E.; Packer, N. H. *Proteomics* "GlycoMod--a software tool for determining glycosylation compositions from mass spectrometric data", 2001, *1*, 340-349.

(19) Cooper, C. A.; Joshi, H. J.; Harrison, M. J.; Wilkins, M. R.; Packer, N. H. *Nucleic Acids Research* "GlycoSuiteDB: a curated relational database of glycoprotein glycan structures and their biological sources. 2003 update", 2003, *31* (1), 511-513.

(20) Domon, B.; Costello, C. E. *Glycoconjugate J.* "A Systematic Nomenclature for Carbohydrate Fragmentations in FABMS/MS of Glycoconjugates", 1988, *5*, 397-409.

(21) Dove, A. In *Nature Biotechnology*, "The bittersweet promise of glycobiology", 2001; Vol. 19, pp 913-917.

(22) Dwek, R. A. *Chem. Rev.* "Glycobiology: Toward Understanding the Function of Sugars", 1996, *96*, 683-720.

(23) Dwek, R. A.; Butters, T. D.; Platt, F. M.; Zitzmann, N. *Nat Rev Drug Discov.* "Targeting glycosylation as a therapeutic approach", 2002, *1* (1), 65-75.

(24) Dziadek, S.; Kunz, H. *Chem Rec.* "Synthesis of tumor-associated glycopeptide antigens for the development of tumor-selective vaccines", 2004, *3* (6), 308-321.

(25) Ethier, M.; Saba, J. A.; Ens, W.; Standing, K. G.; Perreault, H. *Rapid Commun. in Mass Spectrom.* "Automated Structure Assignment of Derivatized Complex N-linked Oligosaccharides from Tandem Mass Spectra", 2002, *16*, 1743-1754.

(26) Ethier, M.; Saba, J. A.; Spearman, M.; Krokhin, O.; Butler, M.; Ens, W.; Standing, K. G.; Perreault, H. *Rapid Commun. Mass Spectrom.* "Application of the StrOligo Algorithm for the Automated Structure Assignment of Complex N-Linked Glycans from Glycoproteins Using Tandem Mass Spectrometry", 2003, *17*, 2713-2720.

(27) Feigenbaum, E. A.; Buchanan, B. G.; Lederberg, J. In *Machine Intelligence 6*; Meltzer, B., Michie, D., Eds.; Edinburgh University Press: Edinburgh, Scotland, 1971, pp 165-190.

(28) Gabius, H.-J.; André, S.; Kaltner, H.; Siebert, H.-C. *Biochim Biophys Acta.* "The sugar code: functional lectinomics", 2002, *1572*, 165-177.

(29) Gabius, H.-J.; Siebert, H.-C.; André, S.; Jiménez-Barbero, J.; Rüdiger, H. *ChemBioChem* "Chemical Biology of the Sugar Code", 2004, *5*, 740-764.

(30) Gaucher, S. P.; Cancilla, M. T.; Phillips, N. J.; Gibson, B. W.; Leary, J. A. *Biochemistry* "Mass spectral characterization of lipooligosaccharides from Haemophilus influenzae 2019", 2000, *39* (40), 12406-12414.

(31) Gaucher, S. P.; Morrow, J.; Leary, J. A. *Anal. Chem.* "STAT: A Saccharide Topology Analysis Tool Used in Combination with Tandem Mass Spectrometry", 2000, *72*, 2331-2336.

(32) Goldberg, D.; Sutton-Smith, M.; Paulson, J.; Dell, A. *Proteomics* "Automatic annotation of matrix-assisted laser desorption/ionization N-glycan spectra." 2005, *4*, 865-875.

(33) Hanneman, A.; Reinhold, V. *Glycobiology* "Abundant and Unusual N-Linked Glycans from the Eukaryote, C. elegans (Abstract 280)", 2003, *13* (11), 899-900.

(34) Hanneman, A.; Singh, S.; Zhang, H.; Reinhold, V., 51st ASMS Conference, "Unraveling Isobaric C. elegans Glycomers: Molecular Disassembly (MS^n) and Structural Continuity (Abstract TPB 031)", Montreal, Quebec, Canada, June 8-12, 2003.

(35) Hanneman, A. J.; Reinhold, V., Joint Meeting of The Society for Glycobiology and The Japanese Society for Carbohydrate Research, "Structural Diversity of C. elegans Glycome (Abstract 252)", Honolulu, Hawaii, Nov. 17-20, 2004.

(36) Harvey, D. J. *Mass Spectrom Rev.* "Matrix-assisted laser desorption/ionization mass spectrometry of carbohydrates", 1999, *18* (6), 349-450.

(37) Harvey, D. J.; Wing, D. R.; Küster, B.; Wilson, I. G. H. *J. Am. Soc. for Mass Spec.* "Composition of N-linked carbohydrates from ovalbumin and co-purified glycoproteins", 2000, *11*, 564-571.

(38) Hedrick, J. L.; Nishihara, T. *J Electron Microsc Tech.* "Structure and function of the extracellular matrix of anuran eggs", 1991, *17* (3), 319-335.

(39) Hokke, C. H.; Deedler, A. M. *Glycoconj J.* "Schistosome glycoconjugates in host-parasite interplay", 2001, *18* (8), 573-587.

(40) Hooper, L. V.; Gordon, J. I. *Glycobiology* "Glycans as legislators of host-microbial interactions: spanning the spectrum from symbiosis to pathogenicity", 2001, *11* (2), 1R-10R.

(41) Huby, R. D.; Dearman, R. J.; Kimber, I. *Toxicol Sci.* "Why are some proteins allergens?" 2000, *55* (2), 235-246.

(42) Ioffe, E.; Stanley, P. *Proc Natl Acad Sci U S A.* "Mice lacking N-acetylglucosaminyltransferase I activity die at mid-gestation, revealing an essential role for complex or hybrid N-linked carbohydrates", 1994, *91* (2), 728-732.

(43) Jaeken, J.; Matthijs, G. *Annual Review of Genomics and Human Genetics* "Congenital disorders of glycosylation", 2001, *2*, 129-151.

(44) Jeyakumar, M.; Butters, T. D.; Dwek, R. A.; Platt, F. M. *Neuropathol Appl Neurobiol.* "Glycosphingolipid lysosomal storage diseases: therapy and pathogenesis", 2002, *28* (5), 343-357.

(45) Joshi, H. J.; Harrison, M. J.; Schulz, B. L.; Cooper, C. A.; Packer, N. H.; Karlsson, N. G. *Proteomics* "Development of a mass fingerprinting tool for automated interpretation of oligosaccharide fragmentation data", 2004, *4*, 1650-1664.

(46) Kannagi, R. *Curr Opin Struct Biol* "Regulatory roles of carbohydrate ligands for selectins in the homing of lymphocytes", 2002, *12* (5), 599-608.

(47) Khoo, K. H.; Dell, A. *Adv Exp Med Biol.* "Glycoconjugates from parasitic helminths: structure diversity and immunobiological implications", 2001, *491*, 185-205.

(48) Koeller, K. M.; Wong, C.-H. *Nature Biotechnology* "Emerging Themes in Medicinal Glycoscience", 2000, *18*, 835-841.

(49) König, S.; Leary, J. A. *J. Am. Soc. for Mass Spec.* "Evidence for linkage position determination in cobalt coordinated pentasaccharides using ion trap mass spectrometry", 1998, *9* (11), 1125-1134.

(50) Küster, B.; Naven, T. J.; Harvey, D. J. *J Mass Spectrom.* "Rapid approach for sequencing neutral oligosaccharides by exoglycosidase digestion and matrix-assisted laser desorption/ionization time-of-flight mass spectrometry", 1996, *31* (10), 1131-1140.

(51) Laine, R. A. *Glycobiology* "A calculation of all possible oligosaccharide isomers both branched and linear yields 1.05 x 10^{12} structures for a reducing hexasaccharide: the Isomer Barrier to development of single-method saccharide sequencing or synthesis systems." 1994, *4* (6), 759-767.

(52) Lapadula, A. J. "GlySpy and the Oligosaccharide Subtree Constraint Algorithm (OSCAR): A Computational Approach to Sequencing Glycans", Technical Report, Dept. of Comp. Sci., Univ. of New Hampshire 2004.

(53) Lapadula, A. J.; Ashline, D. J.; Zhang, H.; Reinhold, V., 54th ASMS Conference on Mass Spectrometry, "Automated Detection of Glycan Isobars with the Bioinformatics Tool GlySpy (Poster ThP 295)", Seattle, WA, May 28 - June 1, 2006.

(54) Lapadula, A. J.; Hatcher, P. J.; Hanneman, A. J.; Ashline, D. J.; Zhang, H.; Reinhold, V. N. *Anal. Chem.* "Congruent Strategies for Carbohydrate Sequencing. 3. OSCAR: An Algorithm for Assigning Oligosaccharide Topology from MS^n Data", 2005, *77* (19), 6271-6279.

(55) Leavell, M. D.; Leary, J. A.; Yamasaki, R. *J. Am. Soc. for Mass Spec.* "Mass Spectrometric Strategy for the Characterization of Lipooligosaccharides from Neisseria gonorrhoeae 302 Using FTICR", 2002, *13*, 571-576.

(56) Lederberg, J., Proceedings of ACM Conference on the History of Medical Informatics, "How DENDRAL Was Conceived and Born", Bethesda, Maryland, United States; 5-19.

(57) Lindsay, R. K.; Feigenbaum, E. A.; Buchanan, B. G.; Lederberg, J. *Applications of Artificial Intelligence for Chemical Inference: The Dendral Project*; McGraw-Hill, Inc.: New York, NY, USA, 1980.

(58) Lo-Man, R.; Vichier-Guerre, S.; Perraut, R.; Deriaud, E.; Huteau, V.; BenMohamed, L.; Diop, O. M.; Livingston, P. O.; Bay, S.; Leclerc, C. *Cancer Res.* "A fully synthetic therapeutic vaccine candidate targeting carcinoma-associated Tn carbohydrate antigen induces tumor-specific antibodies in nonhuman primates", 2004, *64* (14), 4987-4994.

(59) Lowe, J. B.; Marth, J. D. *Annual Rev. Biochem.* "A genetic approach to mammalian glycan function", 2001, *72*, 643-691.

(60) Maeder, T. In *Scientific American*, "Sweet Medicines", 2002.

(61) Marchal, I.; Golfier, G.; Dugas, O.; Majed, M. *Biochemie* "Bioinformatics in glycobiology", 2003, *85*, 75-81.

(62) McLafferty, F. W. *Interpretation of Mass Spectra*, 2nd ed.; W. A. Benjamin: Reading, MA, 1973.

(63) Mozingo, N. M.; Hedrick, J. L. *Developmental Bio* "Distribution of lectin binding sites in Xenopus laevis egg jelly", 1999, *210* (2), 428-439.

(64) Muhlecker, W.; Gulati, S.; McQuillen, D. P.; Ram, S.; Rice, P. A.; Reinhold, V. N. *Glycobiology* "An essential saccharide binding domain for the mAb 2C7 established for Neisseria gonorrhoeae LOS by ES-MS and MS^n." 1999, *9* (2), 157-171.

(65) Nomenclature Committee of the Consortium for Functional Glycomics "Symbol and Text Nomenclature for Representation of Glycan Structure", 2004. http://glycomics.scripps.edu/CFGnomenclature.pdf

(66) Nyame, A. K.; Kawar, Z. S.; Cummings, R. D. *Arch Biochem Biophys* "Antigenic glycans in parasitic infections: implications for vaccines and diagnostics", 2004, *426* (2), 182-200.

(67) Ono, M.; Hakomori, S. *Glycoconjugate Journal* "Glycosylation defining cancer cell motility and invasiveness", 2004, *20*, 71-78.

(68) Parodi, A. J. *Ann. Rev. Biochem.* "Protein glucosylation and its role in protein folding", 2000, *69*, 69-93.

(69) Platt, F. M.; Jeyakumar, M.; Andersson, U.; Heare, T.; Dwek, R. A.; Butters, T. D. *Philos Trans R Soc Lond B Biol Sci.* "Substrate reduction therapy in mouse models of the glycosphingolipidoses", 2003, *358* (1433), 947-954.

(70) Rademacher, T. W.; Parekh, R. B.; Dwek, R. A. *Ann. Rev. Biochem.* "Glycobiology", 1988, *57*, 785-838.

(71) Reinhold, V.; Singh, S.; Zhang, H.; Hanneman, A., Joint Meeting of The Society for Glycobiology and The Japanese Society for Carbohydrate Research, "De novo MS^n Sequencing with Contiguous Glycan Segments (Abstract 490)", Honolulu, Hawaii, Nov. 17-20, 2004.

(72) Reinhold, V. N.; Reinhold, B. B.; Chan, S. *Meth. In Enzym.* "Carbohydrate sequence analysis by electrospray ionization-mass spectrometry", 1996, *271*, 377-402.

(73) Reinhold, V. N.; Reinhold, B. B.; Costello, C. E. *Anal. Chem.* "Carbohydrate Molecular Weight Profiling, Sequence, Linkage, and Branching Data: ES-MS and CID", 1995, *67*, 1772-1784.

(74) Russell, S.; Norvig, P. *Artifical Intelligence: A Modern Approach*, 2 ed.; Prentice Hall: Upper Saddle River, New Jersey, 2003.

(75) Sheeley, D. M.; Reinhold, V. N. *Anal. Chem.* "Structural characterization of carbohydrate sequence, linkage, and branching in a quadrupole ion trap mass spectrometer: Neutral oligosaccharides and N-Linked glycans", 1998, *70*, 3053-3059.

(76) Singh, S.; Reinhold, V. N., Proceedings of 8th Annual Conference of the Society for Glycobiology, "Glycan Disassembly by MS^n: Linkage, Branching and Monomer Identification (Abstract 80)", San Diego, CA, USA, Dec 3-6, 2003.

(77) Singh, S.; Reinhold, V. N.; Bennion, B.; Levery, S. B., Proceedings of 8th Annual Conference of the Society for Glycobiology, "Application of ion trap MS^n strategies to structure elucidation of diverse glycosylinositols derived from fungal glycosphingolipids (Abstract 5)", San Diego, CA, USA, Dec 3-6, 2003.

(78) Stanley, P.; Ioffe, E. *FASEB J.* "Glycosyltransferase mutants: key to new insights in glycobiology", 1995, *9* (14), 1436-1444.

(79) Stephan, M. M. In *The Scientist*, "Sugars Get an 'Ome of their Own", 2004; Vol. 18.

(80) Svennerholm, L. *J. of Neurochemistry* "Chromatographic separation of human brain gangliosides", 1963, *10*, 613-623.

(81) Tang, H.; Mechref, Y.; Novotny, M. V. *Bioinformatics* "Automated interpretation of MS/MS spectra of oligosaccharides", 2005, *21* (Suppl. 1), i431-i439.

(82) Tseng, K.; Hedrick, J. L.; Lebrilla, C. B. *Anal. Chem.* "Catalog-library approach for the rapid and sensitive structural elucidation of oligosaccharides", 1999, *71*, 3747-3754.

(83) Tseng, K.; Xie, Y.; Seeley, J.; Hedrick, J. L.; Lebrilla, C. B. *Glycoconjugate J.* "Profiling with structural elucidation of the neutral and anionic O-linked oligosaccharides in the egg jelly coat of Xenopus laevis by Fourier transform mass spectrometry", 2001, *18*, 309-320.

(84) Turner, M. S.; McKolanis, J. R.; Ramanathan, R. K.; Whitcomb, D. C.; Finn, O. J. *Cancer Chemother Biol Response Modif.* "Mucins in gastrointestinal cancers", 2003, *21*, 259-274.

(85) Van den Steen, P.; Rudd, P. M.; Dwek, R. A.; Opdenakker, G. *Crit Rev Biochem Mol Biol.* "Concepts and principles of O-linked glycosylation", 1998, *33* (3), 151-208.

(86) Various In *Science*, "Carbohydrates and Glycobiology (Special Report)", 2001; Vol. 291, pp 2337-2378.

(87) Varki, A. *Glycobiology* "Biological roles of oligosaccharides: all of the theories are correct", 1993, *3* (2), 97-130.

(88) Varki, A.; Cummings, R.; Esko, J.; Freeze, H.; Hart, G.; Marth, J., Eds. *Essentials of Glycobiology*; Cold Spring Harbor Laboratory Press: New York, 1999.

(89) Viseux, R.; de Hoffman, E.; Domon, B. *Anal. Chem.* "Structural Assignment of Permethylated Oligosaccharide Subunits Using Sequential Tandem Mass Spectrometry", 1998, *70*, 4951-4959.

(90) von der Lieth, C.-W.; Lütteke, T.; Frank, M. *Biochimica et Biophysica Acta* "The role of informatics in glycobiology research with special emphasis on automatic interpretation of MS spectra", 2006, *1760*, 568-577.

(91) Vosseller, K.; Wells, L.; Hart, G. W. *Biochemie* "Nucleocytoplasmic O-glycosylation: O-GlcNAc and functional proteomics", 2001, *83* (7), 575-581.

(92) Walsh, G. *Nature Biotechnology* "Biopharmaceutical benchmarks—2003", 2003, *21*, 865-870.

(93) Weiskopf, A. S.; Vouros, P.; Harvey, D. J. *Rapid Commun. in Mass Spectrom.* "Characterization of Oligosaccharide Composition and Structure by Quadrupole Ion Trap Mass Spectrometry", 1997, *11*, 1493–1504.

(94) Xie, Y.; Tseng, K.; Lebrilla, C. B.; Hedrick, J. L. *J. Am. Soc. for Mass Spec.* "Targeted use of exoglycosidase digestion for the structural elucidation of neutral O-linked oligosaccharides", 2001, *12* (8), 877-884.

(95) Zhang, H.; Reinhold, V., Proceedings of 8th Annual Conference of the Society for Glycobiology, "Composition to Sequence: A Novel Computational Approach to Support MS^n Carbohydrate Sequencing (Abstract 81)", San Diego, CA, USA, Dec 3-6, 2003.

(96) Zhang, H.; Singh, S.; Reinhold, V. *Anal. Chem.* "Congruent Strategies for Carbohydrate Sequencing: 2. FragLib: An MS^n Spectral Library", 2005, *77* (19), 6263-6270.

(97) Zhang, H.; Singh, S.; Reinhold, V., Joint Meeting of The Society for Glycobiology and The Japanese Society for Carbohydrate Research, "Glycan Characterization using a MS^n Fragment Fingerprint Library (Abstract 491)", Honolulu, Hawaii, Nov. 17-20, 2004.